CERN-LHCC-2005-018
ALICE TDR 012
15 June 2005

A L I C E

Technical Design Report

of the

Computing

Cover design by CERN Desktop Publishing Service.

Visualisation of ALICE Experiment as simulated by AliRoot.

Printed at CERN
June 2005.

ISBN 92-9083-247-9

ALICE Collaboration

Alessandria, Italy, Facoltà di Scienze dell'Università and INFN:
P. Cortese, G. Dellacasa, L. Ramello and M. Sitta.

Aligarh, India, Physics Department, Aligarh Muslim University:
N. Ahmad, S. Ahmad, T. Ahmad, W. Bari, M. Irfan and M. Zafar.

Athens, Greece, University of Athens, Physics Department:
A. Belogianni, P. Ganoti, A. Petridis, F. Roukoutakis, M. Spyropoulou-Stassinaki and M. Vassiliou.

Bari, Italy, Dipartimento di Fisica dell'Università and Sezione INFN:
G.E. Bruno, M. Caselle, D. Di Bari, D. Elia, R.A. Fini, B. Ghidini, V. Lenti, V. Manzari, E. Nappi, F. Navach, C. Pastore, V. Paticchio, F. Posa, R. Santoro and I. Sgura.

Bari, Italy, Politecnico and Sezione INFN:
F. Corsi, D. De Venuto, U. Fratino and C. Marzocca.

Beijing, China, China Institute of Atomic Energy:
X. Li, Z. Liu, S. Lu, Z. Lu, Q. Meng, B. Sa, J. Yuan, J. Zhou and S. Zhou.

Bergen, Norway, Department of Physics, University of Bergen:
J. Alme, S. Bablok, A. Klovning, J. Nystrand, B. Pommeresche, M. Richter, D. Röhrich, K. Ullaland and H. Yang.

Bergen, Norway, Bergen University College, Faculty of Engineering:
H. Helstrup K. Hetland and K. Røed.

Bhubaneswar, India, Institute of Physics:
R.K. Choudhury, A.K. Dubey, D.P. Mahapatra, D. Mishra, S.C. Phatak and R. Sahoo.

Birmingham, United Kingdom, School of Physics and Space Research, University of Birmingham:
D. Evans, G.T. Jones, P. Jovanović, A. Jusko, J.B. Kinson, R. Lietava and O. Villalobos Baillie.

Bologna, Italy, Dipartimento di Fisica dell'Università and Sezione INFN:
A. Alici, S. Antinori, P. Antonioli, S. Arcelli, G. Bari, M. Basile, L. Bellagamba, D. Boscherini, A. Bruni, G. Bruni, G. Cara Romeo, L. Cifarelli, F. Cindolo, M. Corradi, D. Falchieri, A. Gabrielli, E. Gandolfi, P. Giusti, D. Hatzifotiadou, G. Laurenti, M.L. Luvisetto, A. Margotti, M. Masetti, R. Nania, F. Noferini, F. Palmonari, A. Pesci, A. Polini, G. Sartorelli, E. Scapparone, G. Scioli, G.P. Vacca, G. Valenti, G. Venturi, M.C.S. Williams, C. Zampolli and A. Zichichi.

Bratislava, Slovakia, Comenius University, Faculty of Mathematics, Physics and Informatics:
V. Černý, R. Janik, L. Lúčan, M. Pikna, J. Pišút, N. Pišútová, B. Sitár, P. Strmeň, I. Szarka and M. Zagiba.

Bucharest, Romania, National Institute for Physics and Nuclear Engineering:
C. Aiftimiei, V. Catanescu, M. Duma, C.I. Legrand, D. Moisa, M. Petrovici and G. Stoicea.

Budapest, Hungary, KFKI Research Institute for Particle and Nuclear Physics, Hungarian Academy of Sciences:
E. Dénes, Z. Fodor, T. Kiss, G. Pálla and J. Zimányi.

Cagliari, Italy, Dipartimento di Fisica dell'Università and Sezione INFN:
I. Atanassov S. Basciu, C. Cicalo, A. De Falco, M. Floris, A. Masoni, D. Mura G. Puddu, S. Serci, E. Siddi and G. Usai.

Catania, Italy, Dipartimento di Fisica dell'Università and Sezione INFN:
A. Badalà, R. Barbera, G. Lo Re, A. Palmeri, G.S. Pappalardo, A. Pulvirenti and F. Riggi.

CERN, Switzerland, European Organization for Nuclear Research:
G. Anelli, I. Augustin[1)], A. Augustinus, J. Baechler, J.A. Belikov[2)], L. Betev, A. Boccardi, R. Brun, M. Burns, P. Buncic[3)], I. Cali[4)], R. Campagnolo, M. Campbell, F. Carena, W. Carena, F. Carminati, N. Carrer, S. Chapeland, C. Cheshkov, P. Chochula, P. Christianssen, P. Christakoglou, A. Colla[5)], J. Conrad, B.F. Correia Belbute, M. Davenport, G. De Cataldo[6)], J. de Groot, A. Di Mauro, R. Divià, C. Engster, S. Evrard, C.W. Fabjan, F. Formenti, U. Fuchs, A. Gallas-Torreira, A. Garcia Lopez A. Gheata, M. Gheata[7)], C. Gonzalez-Gutierrez, R. Grosso[5)], M. Gruwe[8)], H.-A. Gustafsson[9)], H. Hoedelmoser, P. Hristov, M. Ivanov, P. Jacobs[10)], L. Jirden, A. Junique, S. Kapusta, W. Kickinger, W. Klempt, A. Kluge, T. Kuhr[11)], L. Leistam, J.P. Lo, M. Lopez Noriega, C. Lourenço, I. Makhlyueva[12)], J.-C. Marin, P. Martinengo, D. Meunier-Picard, M. Meoni[13)], M. Morel, A. Morsch, B. Mota, H. Muller, L. Musa, P. Nilsson, D. Nouais[14)], F. Osmic, D. Perini, A. Peters[3)], V. Pinto Morais, S. Popescu[15)], F. Rademakers, J.-P. Revol, P. Riedler, W. Riegler, K. Šafařík, P. Saiz, K. Schossmaier, J. Schukraft, Y. Schutz[16)], P.A. Silva Loureiro, C. Soos[17)], G. Stefanini, D. Swoboda, M. Tadel, H. Taureg, M. Tavlet, P. Tissot-Daguette, C. Torcato de Matos, P. Vande Vyvre and J.-P. Vanuxem.

Chandigarh, India, Physics Department, Panjab University:
M.M. Aggarwal, A.K. Bhati, A. Kumar, M. Sharma and G. Sood.

Clermont-Ferrand, France, Laboratoire de Physique Corpusculaire (LPC), IN2P3-CNRS and Université Blaise Pascal:
A. Baldit, V. Barret, N. Bastid, G. Blanchard, J. Castor, P. Crochet, F. Daudon, A. Devaux, P. Dupieux, P. Force, B. Forestier, F. Guerin, R. Guernane, C. Insa, F. Jouve, J. Lecoq, F. Manso, P. Rosnet, L. Royer, P. Saturnini, G. Savinel, G. Stoicea and F. Yermia.

Columbus, U.S.A., Department of Physics, Ohio State University:
T.J. Humanic, I.V. Kotov, M. Lisa, B.S. Nilsen and D. Truesdale.

Columbus, U.S.A., Ohio Supercomputer Centre:
D. Johnson.

Copenhagen, Denmark, Niels Bohr Institute:
I. Bearden, H. Bøggild, C.H. Christensen, J.J. Gaardhøje, K. Gulbrandsen, B.S. Nielsen and G. Renault.

Cracow, Poland, Henryk Niewodniczanski Institute of Nuclear Physics, High Energy Physics Department:
J. Bartke, E. Gładysz-Dziaduś, E. Kornaś, M. Kowalski, A. Rybicki and A. Wróblewski[18)].

Darmstadt, Germany, Gesellschaft für Schwerionenforschung (GSI):
A. Andronic[7)], D. Antonczyk, E. Badura, R. Bailhache E. Berdermann, P. Braun-Munzinger, O. Busch, M. Ciobanu[7)], P. Foka, U. Frankenfeld, C. Garabatos, H. Gutbrod, C. Lippmann, P. Malzacher, A. Marin, D. Miśkowiec, S. Radomski, A. Sandoval, H.R. Schmidt, K. Schwarz, S. Sedykh, R.S. Simon, D. Soyk, H. Stelzer, G. Tziledakis, D. Vranic and J. Wiechula.

Darmstadt, Germany, Institut für Kernphysik, Technische Universität:
U. Bonnes, I. Kraus and H. Oeschler.

Frankfurt, Germany, Institut für Kernphysik, Johann Wolfgang Goethe-Universität:
J. Berger, A. Billmeier, C. Blume, T. Dietel, D. Flierl, M. Gaździcki, Th. Kolleger, S. Lange, C. Loizides, R. Renfordt, R. Stock and H. Ströbele.

Gatchina, Russia, St. Petersburg Nuclear Physics Institute:
Ya. Berdnikov, A. Khanzadeev, N. Miftakhov, V. Nikouline, V. Poliakov, E. Rostchine, V. Samsonov, O. Tarasenkova, V. Tarakanov and M. Zhalov.

Havana, Cuba, Centro de Aplicaciones Technlógicas y Desarrollo Nuclear (CEADEN):
E. Lopez Torres, A. Abrahantes Quintana and R. Diaz Valdes.

Heidelberg, Germany, Kirchhoff Institute for Physics:
V. Angelov, M. Gutfleisch, V. Lindenstruth, R. Panse, C. Reichling, R. Schneider, T. Steinbeck H. Tilsner and A. Wiebalck.

Heidelberg, Germany, Physikalisches Institut, Ruprecht-Karls Universität:
C. Adler, J. Bielčiková, D. Emschermann, P. Glässel, N. Herrmann, Th. Lehmann, W. Ludolphs, T. Mahmoud, J. Milosevic, K. Oyama, V. Petráček, M. Petrovici, I. Rusanov, R. Schicker, H.C. Soltveit, J. Stachel, M. Stockmeier, B. Vulpescu, B. Windelband and S. Yurevich.

Hiroshima, Japan, University of Hiroshima*):
T. Sugitate.

Jaipur, India, Physics Department, University of Rajasthan:
S. Bhardwaj, R. Raniwala and S. Raniwala.

Jammu, India, Physics Department, Jammu University:
S.K. Badyal, A. Bhasin, A. Gupta, V.K. Gupta, S. Mahajan, L.K. Mangotra, B.V.K.S. Potukuchi and S.S. Sambyal.

JINR, Russia, Joint Institute for Nuclear Research:
P.G. Akichine, V.A. Arefiev, Ts. Baatar[19)], B.V. Batiounia, V.F. Chepurnov, S.A. Chernenko, V.K. Dodokhov, O.V. Fateev, A.G. Fedunov, M. Haiduc[20)], D. Hasegan[20)], V.G. Kadychevsky, G. Kharadze[21)], B. Khurelbaatar[19)], E.K. Koshurnikov, V.L. Lioubochits, V.I. Lobanov, L.V. Malinina, Y.I. Minaev, M. Nioradze[22)], P.V. Nomokonov, Y.A. Panebrattsev, V.N. Penev, V.G. Pismennaya, T.A. Pocheptsov, I. Roufanov, G.S. Shabratova, V. Shestakov[23)], A.I. Shklovskaya, A.S. Sorin, M.K. Suleimanov, Y. Tevzadze[22)], R. Togoo[19)], A.S. Vodopianov, V.I. Yurevich, Y.V. Zanevsky, S.A. Zaporojets and A.I. Zinchenko.

Jyväskylä, Finland, Department of Physics, University of Jyväskylä and Helsinki Institute of Physics:
J. Äysto, M. Bondila, V. Lyapin, M. Oinonen, T. Malkiewicz, V. Ruuskanen, H. Seppänen, W. Trzaska and S. Yamaletkinov.

Kangnung, Republic of Korea, Kangnung National University:
H.T. Jung, W. Jung, D.-W. Kim, H.N. Kim, J.S. Kim, K.S. Lee and S.-C. Lee

Karlsruhe, Germany, Institut für Prozessdatenverarbeitung und Elektronik (IPE)*):
T. Blank and H. Gemmeke.

Kharkov, Ukraine, National Scientific Centre, Kharkov Institute of Physics and Technology:
G.L. Bochek, A.N. Dovbnya, V.I. Kulibaba, N.I. Maslov, S.V. Naumov, V.D. Ovchinnik, S.M. Potin and A.F. Starodubtsev.

Kharkov, Ukraine, Scientific and Technological Research Institute of Instrument Engineering:
V.N. Borshchov, O. Chykalov, L. Kaurova, S.K. Kiprich, L. Klymova, O.M. Listratenko, N. Mykhaylova, M. Protsenko, O. Reznik and V.E. Starkov.

Kiev, Ukraine, Department of High Energy Density Physics, Bogolyubov Institute for Theoretical Physics, National Academy of Sciences of Ukraine:
O. Borysov, I. Kadenko, Y. Martynov, S. Molodtsov, O. Sokolov, Y. Sinyukov and G. Zinovjev.

Kolkata, India, Saha Institute of Nuclear Physics:
P. Bhattacharya, S. Bose, S. Chattopadhyay, N. Majumdar, S. Mukhopadhyay, A. Sanyal, S. Sarkar, P. Sen, S.K. Sen, B.C. Sinha and T. Sinha.

Kolkata, India, Variable Energy Cyclotron Centre:
Z. Ahammed, P. Bhaskar, S. Chattopadhyay, D. Das, S. Das, M.R. Dutta Majumdar, M.S. Ganti, P. Ghosh, B. Mohanty, T.K. Nayak, P.K. Netrakanti, S. Pal, R.N. Singaraju, V. Singhal, B. Sinha, M.D. Trivedi and Y.P. Viyogi.

Köln, Germany, University of Applied Sciences Cologne, Communications Engineering[*)]:
G. Hartung and T. Krawutschke.

Košice, Slovakia, Institute of Experimental Physics, Slovak Academy of Sciences and Faculty of Science, P.J. Šafárik University:
J. Bán, M. Bombara, A. Dirner, S. Fedor, M. Hnatič, I. Králik, A. Kravčáková, F. Kriváň, M. Krivda, G. Martinská, B. Pastirčák, L. Šándor, J. Urbán and J. Vrláková.

Legnaro, Italy, Laboratori Nazionali di Legnaro:
M. Cinausero, E. Fioretto, G. Prete, R.A. Ricci and L. Vannucci.

Lisbon, Portugal, Departamento de Fìsica, Instituto Superior Tècnico:
P. Branco, R. Carvalho, J. Seixas and R. Vilela Mendes.

Lund, Sweden, Division of Experimental High Energy Physics, University of Lund:
A. Oskarsson, L. Osterman, I. Otterlund and E.A. Stenlund.

Lyon, France, Institut de Physique Nucléaire de Lyon (IPNL), IN2P3-CNRS and Université Claude Bernard Lyon-I:
B. Cheynis, L. Ducroux, J.Y. Grossiord, A. Guichard, P. Pillot, B. Rapp and R. Tieulent.

Mexico City and Merida, Mexico, Centro de Investigacion y de Estudios Avanzados del IPN, Universidad Nacional Autonoma de Mexico, Instituto de Ciencias Nucleares, Instituto de Fisica:
J.R. Alfaro Molina, A. Ayala, E. Belmont Moreno, J.G. Contreras, E. Cuautle, J.C. D'Olivo, I. Dominguez, A. Flores, G. Herrera Corral, I. Leon Monzon, M. Linares, J. Martinez Castro, M.I. Martinez, A. Martinez Davalos, A. Menchaca-Rocha, L.M. Montano Zetina, L. Nellen, G. Paic[24)], J. del Pino, P. Reyes, A. Sandoval, J. Solano, M.A. Vargas and A. Zepeda.

Moscow, Russia, Institute for Nuclear Research, Academy of Science:
V.A. Feshcenko, M.B. Golubeva, V.G. Gorlychev, F.F. Guber, O.V. Karavichev, T.L. Karavicheva, E.V. Karpechev, A.B. Kurepin, A.I. Maevskaya, V.V. Marin, I.A. Pshenichnov, V.I. Razin, A.I. Rechetin, K.A. Shileev and N.S. Topil'skaia.

Moscow, Russia, Institute for Theoretical and Experimental Physics:
A.N. Akindinov, V. Golovine, A.B. Kaidalov, M.M. Kats, I.T. Kiselev, S.M. Kisselev, E. Lioublev, M. Martemianov, A.N. Martemiyanov, P.A. Polozov, V.S. Serov, A.V. Smirnitski, M.M. Tchoumakov, I.A. Vetlitski, K.G. Volochine, L.S. Vorobiev and B.V. Zagreev.

Moscow, Russia, Russian Research Center Kurchatov Institute :
D. Aleksandrov, V. Antonenko, S. Beliaev, S. Fokine, M. Ippolitov, K. Karadjev, V. Lebedev, V.I. Manko, T. Moukhanova, A. Nianine, S. Nikolaev, S. Nikouline, O. Patarakine, D. Peressounko, I. Sibiriak, A. Tsvetkov, A. Vasiliev, A. Vinogradov, M. Volkov and I. Yushmanov.

Moscow, Russia, Moscow Engineering Physics Institute:
V.A. Grigoriev, V.A. Kaplin and V.A. Loginov.

Mumbai, India, Indian Institute of Technology,*):
B.K. Nandi and R. Varma.

Mumbai, India Bhabha Atomic Research Centre (BARC), *):
V. Chandratre and V. Kataria.

Münster, Germany, Institut für Kernphysik, Westfälische Wilhelms Universität:
C. Baumann, D. Bucher, R. Glasow, H. Gottschlag, J.F. Groše-Oetringhaus, N. Heine, C. Klein-Bösing, K. Reygers, R. Santo, W. Verhoeven, J. Wessels and A. Wilk.

Nantes, France, Laboratoire de Physique Subatomique et des Technologies Associées (SUBATECH), Ecole des Mines de Nantes, IN2P3-CNRS and Université de Nantes:
L. Aphecetche, J.M. Barbet, G. Conesa-Balbastre, Z. Conesa-del-Valle, J.P. Cussonneau, H. Delagrange, M. Dialinas, C. Finck, B. Erazmus, M. Germain, S. Kabana, F. Lefèvre, L. Luquin, L. Martin, G. Martinez, C. Roy and A. Tournaire.

Novosibirsk, Russia, Budker Institute for Nuclear Physics:
A.R. Frolov and I.N. Pestov.

Oak Ridge, U.S.A., Oak Ridge National Laboratory:
T. Awes.

Omaha, U.S.A., Creighton University:
M. Cherney.

Orsay, France, Institut de Physique Nucléaire (IPNO), IN2P3-CNRS and Université de Paris-Sud:
L. Bimbot, V. Chambert, A. Charpy, M.P. Comets, P. Courtat, S. Drouet, P. Edelbruck, B. Espagnon, I. Hřivnáčová, R. Kunne, Y. Le Bornec, M. Mac Cormick, J. Peyré, J. Pouthas, S. Rousseau, C. Suire and N. Willis.

Oslo, Norway, Department of Physics, University of Oslo:
L. Bravina, G. Løvhøiden, B. Skaali, T.S. Tveter and T. Vik.

Padua, Italy, Dipartimento di Fisica dell'Università and Sezione INFN:
F. Antinori, A. Dainese, R. Dima, D. Fabris, J. Faivre, M. Lunardon, M. Morando, S. Moretto, A. Pepato, E. Quercigh, F. Scarlassara, G. Segato, R. Turrisi and G. Viesti.

Pohang, Republic of Korea, Pohang Accelerator Laboratory:
J. Choi.

Prague, Czech Republic, Institute of Physics, Academy of Science:
A. Beitlerova, J. Mareš, K. Polák and P. Závada.

Prague, Czech Republic, Czech Technical University Faculty of Nuclear Sciences and Physical Engineering:
V. Petráček, M. Pachr and L. Škoda.

Protvino, Russia, Institute for High Energy Physics:
M.Yu. Bogolyubsky, G.V. Khaoustov, I.V. Kharlov, N.G. Minaev, V.S. Petrov, B.V. Polichtchouk, S.A. Sadovsky, V.A. Senko, A.S. Soloviev, P.V. Stolpovsky and V.A. Victorov.

Puebla, Mexico, Benemerita Universidad Autonoma de Puebla:
A. Fernandez Tellez, E. Gamez Flores, R. Lopez, G. Tejeda and S. Vergara

Řež u Prahy, Czech Republic, Academy of Sciences of Czech Republic, Nuclear Physics Institute:
D. Adamová, S. Kouchpil, V. Kouchpil, A. Kugler, M. Šumbera and V. Wagner.

Rome, Italy, Dipartimento di Fisica dell'Università 'La Sapienza' and Sezione INFN:
S. Di Liberto, M.A. Mazzoni, F. Meddi and G.M. Urciuoli.

Rondebosch, South Africa, University of Cape Town:
J. Cleymans, G. de Vaux, R W. Fearick, A. Szostak and Z.Z. Vilakazi.

Saclay, France, Centre d'Études Nucléaires, DAPNIA:
M. Anfreville, A. Baldisseri, B. Becker, H. Borel, J.-L. Charvet, M. Combet, J. Gosset, P. Hardy, S. Herlant, F. Orsini, Y. Pénichot, H. Pereira, S. Salasca, F.M. Staley and M. Usseglio.

Salerno, Italy, Dipartimento di Fisica 'E.R.Caianiello' dell'Università and Sezione INFN:
A. De Caro, C. Guarnaccia, D. De Gruttola, S. De Pasquale, A. Di Bartolomeo, M. Fusco Girard, G. Grella, M. Guida, G. Romano, S. Sellitto, R. Silvestri and T. Virgili.

Sarov, Russia, Russian Federal Nuclear Center (VNIIEF):
V. Basmanov, D. Budnikov, V. Demanov, V. Ianowski, R. Ilkaev, L. Ilkaeva, A. Ivanov, A. Khlebnikov, A. Kouryakin, S. Nazarenko, V. Pavlov, S. Philchagin, V. Punin, S. Poutevskoi, I. Selin, I. Vinogradov, S. Zhelezov and A. Zhitnik.

Split, Croatia, Technical University of Split, Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture (FESB):
S. Gotovac, E. Mudnic and L. Vidak.

St. Petersburg, Russia, V. Fock Institute for Physics of St. Petersburg State University :
M.A. Braun, G.A. Feofilov, S.N. Igolkin, A.A. Kolojvari, V.P. Kondratiev, P.A. Otyugova, O.I. Stolyarov, T.A. Toulina, F.A. Tsimbal, F.F. Valiev, V.V. Vechernin and L.I. Vinogradov.

Strasbourg, France, Institut de Recherches Subatomiques (IReS), IN2P3-CNRS and Université Louis Pasteur:
J. Baudot, D. Bonnet, J.P. Coffin, B. Hippolyte, C. Kuhn, J.R. Lutz and R. Vernet.

Tokyo, Japan, University of Tokyo,*):
H. Hamagaki and K. Ozawa.

Trieste, Italy, Dipartimento di Fisica dell'Università and Sezione INFN:
V. Bonvicini, O. Borysov, L. Bosisio, M. Bregant, P. Camerini, G. Contin, F. Faleschini, E. Fragiacomo, N. Grion, G. Margagliotti, S. Piano, I. Rachevskaya, A. Rachevski, R. Rui and A. Vacchi.

Turin, Italy, Dipartimenti di Fisica dell'Università and Sezione INFN:
B. Alessandro, R. Arnaldi, G. Batigne, S. Beolé, E. Bruna, P. Cerello, E. Chiavassa, S. Coli, E. Crescio, N. De Marco, A. Ferretti, M. Gagliardi, M. Gallio, L. Gaudichet, R. Gemme, G. Giraudo, P. Giubellino, M. Idzik, S. Martoiu, A. Marzari Chiesa, M. Masera, G. Mazza, P. Mereu, M. Monteno, A. Musso, C. Oppedisano, A. Piccotti, F. Poggio, F. Prino, L. Riccati, A. Rivetti, E. Scomparin, F. Tosello, L. Toscano, G. Travaglia, E. Vercellin, A. Werbrouck and R. Wheadon.

Tsukuba, Japan, University of Tsukuba,*):
Y. Miake and S. Esumi.

Utrecht, The Netherlands, Subatomic Physics Department, Utrecht University and National Institute for Nuclear and High Energy Physics (NIKHEF):
M. Botje, J.J.F. Buskop, A.P. De Haas, R. Kamermans, P.G. Kuijer, G. Nooren, C.J. Oskamp, Th. Peitzmann, E. Simili, R. Snellings, A.N. Sokolov, A. Van Den Brink and N. Van Eijndhoven.

Wako-shi, Japan, Institute of Research, RIKEN*):
H. Enyo, K. Fujiwara, H. Kano and H. Onishi.

Warsaw, Poland, Soltan Institute for Nuclear Studies:
A. Deloff, T. Dobrowolski, K. Karpio, M. Kozlowski, H. Malinowski, K. Redlich[25)], T. Siemiarczuk, G. Stefanek[26)], L. Tykarski and G. Wilk.

Warsaw, Poland, University of Technology, Institute of Physics:
Z. Chajecki, H. Gos, M. Janik, M. Jedynak, A. Kisiel, T.J. Pawlak, W.S. Peryt, J. Pluta, P. Skowronski, M. Slodkowski, P. Szuba and T. Traczyk.

Worms, Germany, University of Applied Sciences Worms, ZTT*):
E.S. Conner and R. Keidel.

Wuhan, China, Institute of Particle Physics, Huazhong Normal University:
X. Cai, X.R. Wang, T. Wu, H.Y. Yang, Z.B. Yin and D.C. Zhou.

Wuhan, China, Huazhong Univ. of Science and Techn. Electronic and Inform. Engineering:
X. Cao and Q. Li.

Yerevan, Armenia, Yerevan Physics Institute:
M. Atayan, A. Grigoryan, S. Grigoryan, H. Gulkanyan, A. Harutyunyan, A. Hayrapetyan, V. Kakoyan, M. Poghosyan and G. Sargsyan.

Zagreb, Croatia, Ruder Bošković Institute:
T. Anticic, K. Kadija and T. Susa.

*) Associate member.
1) Now at Darmstadt, Germany, Gesellschaft für Schwerionenforschung (GSI)
2) On leave from JINR, Dubna, Russia.
3) supported by EGEE
4) On leave from Università del Piemonte Orientale, Alessandria, Italy.
5) On leave from INFN, Italy
6) also at INFN, Bari, Italy
7) On leave from National Institute for Physics and Nuclear Engineering, Bucharest, Romania.
8) Also at GSI, Darmstadt, Germany
9) On leave from Division of Experimental High Energy Physics, University of Lund, Lund, Sweden
10) On leave from Lawrence Berkeley National Laboratory
11) Now at Institut fuer Theoretische Teilchenphysik, Universitaet Karlsruhe
12) Also at ITEP, Moscow, Russia
13) Now at INFN-Firenze
14) Also at Also at INFN, Sezione di Torino, Italy
15) Also at IFIN-HH, HEP Dept., Bucharest, Romania
16) On leave from Laboratoire de Physique Subatomique et des Technologies Associées (SUBATECH), Ecole des Mines de Nantes, IN2P3-CNRS and Université de Nantes, Nantes, France.
17) On leave from Budapest University, Budapest, Hungary.
18) Cracow Technical University, Cracow, Poland.
19) Institute of Physics and Technology, Mongolian Academy of Sciences, Ulaanbaatar, Mongolia.
20) Institute of Space Sciences, Bucharest, Romania.
21) Institute of Physics, Georgian Academy of Sciences, Tbilisi, Georgia.
22) High Energy Physics Institute, Tbilisi State University, Tbilisi, Georgia.
23) Research Centre for Applied Nuclear Physics (RCANP), Dubna, Russia.
24) Department of Physics, Ohio State University, Columbus, U.S.A.
25) Physics Faculty, University of Bielefeld, Belfield, Germany and University of Wroclaw, Wroclaw, Poland.
26) Institute of Physics, Pedagogical University, Kielce, Poland.

Acknowledgements

The Collaboration wishes to thank all the administrative and technical staff involved during the preparation of the Computing TDR and in particular: the ALICE secretariat, M. Connor, U. Genoud, and C. Hervet; the CERN Desktop Publishing Service, in particular S. Leech O'Neale and C. Vanoli; and the CERN printshop.

Summary

The principal objective of this document is to present the computing model of ALICE (A Large Ion Collider Experiment) at the CERN [1] Large Hadron Collider [2] (LHC) and the current estimates of the computing resources needed to implement this model. The content of this document is the result of a long series of consultations and discussions within the ALICE Collaboration and its governing bodies. At the beginning of 2005, a LHCC review [3] examined the computing resources requested by the LHC experiments and found these requests reasonable. As expected and announced at the time of the review, the computing model and the projected computing needs have further evolved as a result of the additional experience gained in processing the data produced by the Data Challenges.

At the time of writing, we are slightly more than two years away from the first collisions at the LHC. This is, however, still a long lapse of time because of the fast pace of evolution in the field of Information Technology. In addition, the anticipated needs for LHC computing are very large. So is the complexity of the environment required to process the data and make optimal use of the available resources. Therefore, the deployment and organisation of the software, of the material and of the human resources needed have to be properly planned. This requirement is particularly critical, since the resources will be distributed in many centres around the world which will have to work together in a coherent way as a single entity.

In consideration of the above, this document contains the appropriate level of detail to support the ALICE requests and guide the collaboration in the implementation and deployment of the ALICE software and computing infrastructure, ideally without basing the ALICE computing strategy on elements that can and will still change in the course of the next few years.

The ALICE offline framework (AliRoot) has been under development since 1998. It has provided inputs for the Technical Design Reports of all ALICE detectors and for the performance and physics studies presented in the ALICE Physics Performance Report [4]. The AliRoot framework is based on Object-Oriented technology and depends on the ROOT framework. Although AliRoot already allows quite detailed and realistic studies of the detector, it is still under intense development.

Advanced code inspection tools have been developed in collaboration with computer science experts, and these have now been deployed in production. They play an important role in maintaining the quality and uniformity of the AliRoot code.

Simulation has so far been performed with the GEANT 3 Monte Carlo transport program through the use of a set of interfaces that allow the transparent implementation of other Monte Carlo transport programs. The interface to the FLUKA Monte Carlo program has also been validated, and it is being used in production. In the near future we plan to upgrade the existing interface to the GEANT 4 Monte Carlo transport program and therefore to discontinue the GEANT 3 program. The geometry is described via a modeller developed jointly by ALICE and the ROOT team.

A large amount of work has been dedicated to reconstruct trajectories and identify particles. These tasks are particularly difficult in the context of heavy-ion collisions, as we expect that such collisions will produce a number of tracks an order of magnitude larger than in proton–proton (pp) collisions and that the occupancy can be as high as 40% in some regions. The results obtained from the simulation in terms of efficiency and contamination are very close to the design parameters. More work is being carried out to consolidate these results in conjunction with the calibration and alignment algorithms, still under development.

The complete design of the condition infrastructure (calibration and alignment) is ready. Pilot implementations have already been achieved for a few detectors. Validation tests will be performed during the Physics Data Challenge 2005.

The development of the visualization application has just started and will be continued over the coming year.

The computing model applies to a ‘Standard Data Taking Year’ (SDTY). During a SDTY, ALICE

will take heavy-ion data for 10^6 effective seconds per year (one month), while for the rest of the time, when the accelerator is active, 10^7 effective seconds, ALICE will collect proton-proton data. During the initial phase, we assume that the effective beam time may be less, and increasing luminosity will progressively become available. However, the exact operation during the first three years, the so-called 'initial running conditions', is being periodically reassessed. Our model takes into account this period through a staging of the deployment of the computing resources, i.e. 20% to be available in 2007 for the first pp run, 40% for the first Pb–Pb pilot run in 2007, and 100% for the first full Pb–Pb run in 2008, even if the nominal luminosity is not yet available. This responds to the requirement to delay the acquisition of hardware as much as possible, every year bringing a reduction in cost that for CPU's can reach of 30–40%.

The computing model for the pp data is similar to that of the other LHC experiments. Data are recorded at a rate of 100 MB/s. They are reconstructed quasi on-line at the CERN Tier 0 facility. In parallel, data are exported to the different Tier 1s outside CERN (hereafter 'external Tier 1s'), to provide two copies of the raw data, one stored at the CERN Tier 0 and another copy shared by all the external Tier 1s. All Tier 1s will have collectively enough resources to perform a second and third reconstruction pass.

For heavy-ion data this model is not viable, as data are recorded at up to 1.25 GB/s. Such a data rate would require a prohibitive amount of resources for quasi real-time processing. ALICE therefore requires that heavy-ion data be reconstructed at the CERN Tier 0 and exported during a period of four months after data taking. Additional reconstruction passes will be performed at the Tier 1s.

It is customary to assume that scheduled analysis will be performed at Tier 1 centres, while unscheduled analysis and simulation will be performed at the Tier 2 centres. On the basis of the experience gained with the Physics Data Challenges, this hierarchical model, based on the MONARC [5] work, may be progressively replaced by a more 'symmetric' model often referred to as the 'cloud model'. In the latter model, the only distinctive features of the Tier 1s, apart from size, are service levels and the commitment to store the data safely, most likely on mass storage systems.

The choice of the model finally adopted will also depend on the functionality and reliability of the Grid middleware. Should the middleware have a limited functionality in deciding where to perform the calculations and where to direct the data, a hierarchical model will be useful in organizing 'by hand' the computing activity. A middleware implementing a set of functionalities closer to the 'Grid vision' could benefit from some more freedom of choice, leading to a usage pattern of the resources similar to the one predicted by the cloud model.

At the time of writing, the functionality of the Grid middleware that will be installed on the LHC Computing Grid [6] (LCG) resources is still evolving. The elaboration of the ALICE computing model has implicitly assumed that a functional middleware will exist, optimizing to some extent the storage and workload distribution. Based on the experience gained with the ALICE-developed AliEn [7] system, it is believed that the application of the cloud model is technically possible. Currently, it is planned to provide the required Grid functionality via a combination of the common Grid services offered on the LCG resources and the ALICE-specific services from AliEn.

To ease the estimation of required resources, each task has been assigned to a specific Tier, in accordance with the MONARC model. Throughout this document the MONARC terminology will be used to discuss the different elements.

Finally, it is important to note that all the information contained in this document is provided to the best of our knowledge. The contents of this document depend on a number of human and technological factors that are in rapid evolution. We anticipate a qualitative as well as quantitative evolution of the ALICE computing model.

The document is organized as follows. Chapter 1 contains a description of the data acquisition system (DAQ) and of the basic parameters of the raw data. These are fundamental inputs for the computing infrastructure and the computing model. Chapter 2 provides an overview of the computing framework together with the condition infrastructure. Chapter 3 describes the ALICE distributed computing en-

vironment and the ALICE experience with the Data Challenges. Chapter 4 describes the simulation infrastructure. Chapter 5 illustrates the reconstruction strategy and the current status of the performance of the algorithms. Chapter 6 contains our current plans for the development of the analysis framework and some prototype implementation. Chapter 7 describes the ALICE computing model and the projected computing needs. Chapter 8 presents the organization and funding structure of the ALICE Computing Project and lists the major milestones of the project.

Contents

ALICE Collaboration **i**

Summary **xi**

Contents **xv**

1 Basic parameters and raw data structure **1**
1.1 Introduction . . . 1
1.2 Raw data structure . . . 1
1.2.1 Trigger, Data Acquisition, and High-Level Trigger systems . . . 1
1.2.2 Raw data format . . . 2
1.2.3 Common data format . . . 3
1.2.4 Central Trigger Processor readout and interaction-record data format . . . 5
1.2.5 High-Level Trigger readout . . . 7
1.2.6 The DATE ROOT recorder . . . 8
1.3 Computing data challenges . . . 8
1.3.1 Introduction . . . 8
1.3.2 Performances and results . . . 10
1.3.3 Future plans . . . 12

2 Overview of the computing framework **13**
2.1 The development of AliRoot . . . 13
2.2 ROOT framework . . . 13
2.3 AliRoot framework . . . 15
2.3.1 The function of AliRoot . . . 15
2.3.2 Principles of AliRoot design . . . 16
2.3.3 Data structure design . . . 17
2.3.4 Offline detector alignment and calibration model . . . 18
2.3.5 Tag database . . . 21
2.3.6 LHC common software . . . 21
2.4 Software Development Environment . . . 22
2.5 Maintenance and distribution of AliRoot . . . 23
2.5.1 Code development tools . . . 24

3 Distributed computing and the Grid **25**
3.1 Introduction . . . 25
3.2 Distributed computing . . . 25
3.3 AliEn, the ALICE interface to the Grid . . . 27
3.4 Future of the Grid in ALICE . . . 30

4 Simulation **33**
4.1 Event generators . . . 33
4.2 Afterburner processors and correlation analysis . . . 36
4.3 Detector response simulation . . . 36
4.3.1 Simulation framework . . . 39
4.3.2 Geometry of structural elements . . . 39
4.3.3 Geometry of detectors . . . 40

4.4 Fast simulation . . . 42
4.5 Event merging and embedding . . . 42

5 Reconstruction **43**
5.1 Organization of the reconstruction code . . . 43
5.2 Track reconstruction in the central detectors . . . 44
5.3 Track reconstruction in the forward muon spectrometer . . . 47
5.4 Charged particle identification . . . 48
5.5 Photon and neutral meson reconstruction in the PHOS . . . 50
5.6 High-Level Trigger reconstruction . . . 51

6 Data analysis **53**
6.1 Introduction . . . 53
6.1.1 The analysis activity . . . 53
6.2 Organization of the data analysis . . . 54
6.3 Infrastructure tools for distributed analysis . . . 55
6.3.1 gShell . . . 55
6.3.2 PROOF – the Parallel ROOT Facility . . . 55
6.4 Analysis tools . . . 56
6.4.1 Statistical tools . . . 56
6.4.2 Calculations of kinematics variables . . . 56
6.4.3 Geometrical calculations . . . 56
6.4.4 Global event characteristics . . . 57
6.4.5 Comparison between reconstructed and simulated parameters . . . 57
6.4.6 Event mixing . . . 57
6.4.7 Analysis of the High-Level Trigger (HLT) data . . . 58
6.4.8 Visualization . . . 58
6.5 Existing analysis examples in AliRoot . . . 58

7 Computing model and capacity requirements **61**
7.1 Introduction . . . 61
7.2 Input parameters . . . 61
7.3 The computing model . . . 62
7.4 CPU requirements . . . 63
7.4.1 Parameter values . . . 63
7.4.2 Raw-data processing strategy . . . 65
7.4.3 Monte Carlo data simulation . . . 66
7.4.4 Data analysis . . . 66
7.5 Storage requirements . . . 68
7.5.1 Permanent storage . . . 68
7.5.2 Transient storage . . . 69
7.6 Network . . . 69
7.6.1 Tier 0 . . . 69
7.6.2 Tier 1 . . . 70
7.6.3 Tier 2 . . . 70
7.7 Ramp-up of resources . . . 70
7.8 Summary . . . 72

8 Computing Project organization and responsibilities **73**
8.1 Computing Project responsibilities . . . 73
8.2 Computing Project organization . . . 73

8.3 Organization of the Core Computing Project 75
8.4 Activities in the Core Computing Project 76
8.5 Institutes participating in the activities 77
8.6 Milestones 77

References **79**

1 Basic parameters and raw data structure

1.1 Introduction

This chapter describes the basic parameters of the ALICE data, used as input to the computing model and resources estimate. A brief description of the ALICE Data Acquisition (DAQ) system is given, together with a discussion on the Computing Data Challenges.

1.2 Raw data structure

1.2.1 Trigger, Data Acquisition, and High-Level Trigger systems

The architecture of the ALICE Data Acquisition and its interfaces with the Trigger and the High-Level Trigger are illustrated in Fig. 1.1 and detailed in the ALICE Technical Design Report on Trigger, Data Acquisition, High Level Trigger, and Control System [1].

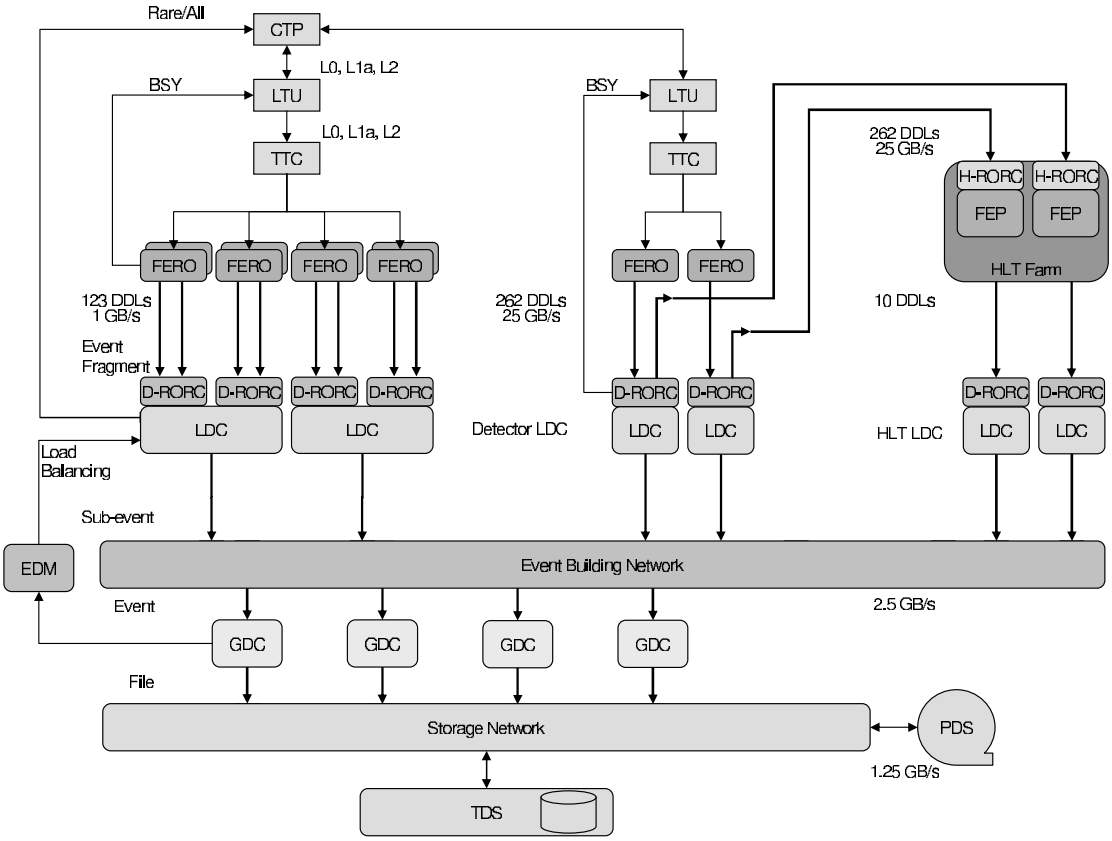

Figure 1.1: DAQ architecture overview.

The detectors receive the trigger signals and the associated information from the Central Trigger Processor (CTP) [2], through a dedicated Local Trigger Unit (LTU) [3] interfaced to a Timing, Trigger and Control (TTC) system [4]. The Front End Read Out (FERO) electronics of the detectors is interfaced to the ALICE Detector Data Links (DDLs). The data produced by the detectors (event fragments) are injected into the DDLs using a common protocol.

At the receiving side of the DDLs there are PCI-based electronic modules, called DAQ Readout Receiver Cards (D-RORCs). The D-RORCs are hosted by the front-end machines (commodity PCs), called Local Data Concentrators (LDCs). Each LDC can handle one or more D-RORCs. The event fragments originated by the various D-RORCs are logically assembled into sub-events in the LDCs.

The CTP receives a busy signal from each detector. This signal can be generated either in the detector electronics or from all the D-RORCs of a detector. The CTP also receives a signal from the DAQ enabling or disabling the most common triggers. It is used to increase the acceptance of rare triggers by reducing the detector dead-time. This signal is function of the buffer occupancy in all the LDCs.

The role of the LDCs is to ship the sub-events to a farm of machines (also commodity PCs) called Global Data Collectors (GDCs), where the whole event is built (from all the sub-events pertaining to the same trigger). The GDCs also feed the Transient Data Storage (TDS) with the events that eventually end up in Permanent Data Storage (PDS). The PDS is managed by the CERN Advanced Storage Manager (CASTOR) [5].

All these hardware elements are driven and controlled by the Data Acquisition and Test Environment (DATE) software framework [6] developed by the ALICE DAQ project. The coherence of the whole project is ensured by this common software framework composed of different layers of modules. A bottom layer includes the memory handling, the process synchronization, and the communication modules. The application layer includes the data-flow applications (detector readout, event building, and data recording). DATE has been used since several years by many ALICE test beams. Most of the ALICE detectors have already realised a complete integration of their readout system with the DDL and the DATE software.

The High Level Trigger (HLT) system [1] receives a copy of all the raw data. The data and decisions generated by HLT are transferred to dedicated LDCs.

1.2.2 Raw data format

The ALICE raw data format is formulated according to the needs of the computing model and following the architecture of the Trigger, DAQ and HLT systems [1].

The raw data format covers the following pieces of data:

- The event fragment is the block of data as sent by the detector electronics over the ALICE DDL. A common data format is defined [7] for all the data blocks transferred by every detector readout electronics over the DDL [8, 9]. This data format contains all the information needed by the DAQ for the sub-event and event building.

  The data format described here is generated by the readout electronics. It covers the mandatory minimum data format for all the data blocks sent over the DDL and provides the information required to identify the data in the DAQ system. The identification of a data block and its processing is based on the format version, the event identification and the trigger information. The corresponding fields of the common data format header are therefore mandatory.

  In addition, three optional fields are added to the common data format header: the block length, the event attributes, and the Region Of Interest (ROI) data. Extensions could be defined in the data themselves.

  The format used for the data generated by the CTP is also presented hereafter.

- The sub-event is the block of data resulting from the merging of one or several event fragments. It is generated by the LDC and is the smallest amount of data that can be monitored in the DAQ system.

- The event is the assembly of all the sub-events pertaining to the same physics interaction. It is generated by the GDC.

1.2.3 Common data format

Data transferred from the front-end electronics to the LDCs is formatted as follows (Fig. 1.2):

- A header describing the associated data block(s), the trigger conditions, the error and status conditions and other information dependent on the readout electronics.
- Zero, one, or more data blocks belonging to the same physics or software trigger may follow the header. The data blocks are optional and may be skipped if there is no valid data associated with a given trigger.

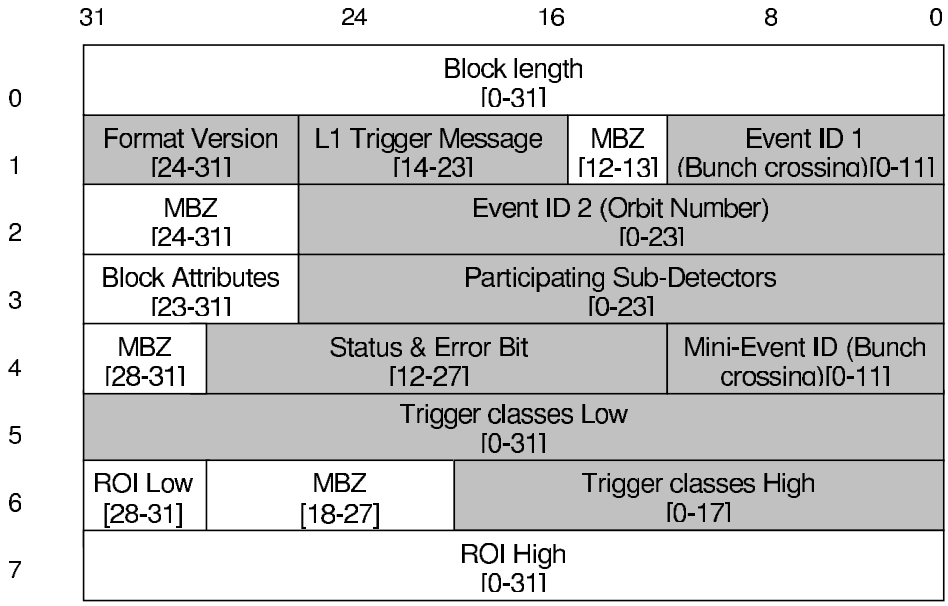

Figure 1.2: Common data format header.

The various fields of the common data format header are either loaded using the data transmitted by the ALICE Trigger system or created locally by the front-end electronics when running without the ALICE Trigger system (e.g. for standalone tests). The presence or absence of the ALICE Trigger system is marked by the trigger information unavailable status bit. When running without the ALICE Trigger system, the information contained in most of the fields is irrelevant.

The fields of Fig. 1.2 marked with a white background (e.g. Block length) are optional and can be left void if their value is not available or not needed. The fields marked in a grey background (e.g. Format Version) are mandatory and are filled with their correct value, as defined at the moment the event is generated. All fields marked MBZ (Must Be Zero) are reserved for future use and are set to zero by the readout electronics. A description of the individual fields of the common data format header follows.

Block Length

The Block Length is an optional field. It can be filled in by the detector readout electronics to indicate the total length of the data block including header and payload(s). The length is expressed in bytes being transferred over the DDL.

Format Version

The Format Version indicates which version of the present data format is used. The presence of this field will provide backward compatibility in case of change or upgrade.

L1 Trigger Message

The L1 Trigger Message consists of selected parts of the L1 trigger word [2]. This information is distributed over the TTC [4] to the detector readout. This field is a direct (bit-by-bit) copy of the first ten data bits of the L1 Data Message Word 1. It contains the information given in Table 1.1.

Table 1.1: L1 Trigger Message fields.

L1 Trigger Message field of DDL header	Data payload of the L1 Data Message Word 1
Bit 23–20	Spare bits
Bit 20	ClT bit
Bit 19–16	RoC[4...1]
Bit 15	ESR bit
Bit 14	L1SwC bit

Event ID

The LHC clock supplies the event identification in ALICE. This clock is distributed to all the detector readout units by the TTC system used as trigger distribution network. The current LHC design foresees 3564 bunches in one orbit. The LHC clock identifies each bunch crossing within an orbit and signals the beginning of a new orbit. Currently the TTC foresees 12 bits for the bunch-crossing number. The Trigger system includes a cyclic counter of 24 bits to count the orbit. This scheme uniquely identifies every bunch crossing in a period of more than 20 minutes ($(2^{24}\text{-}1) \times 88\ \mu$s = 1476 s = 24 min), which is sufficient for this purpose. Further identification is added by the DAQ to uniquely identify one event in a run. The information stored in the Event ID fields (1 and 2) is transmitted by the CTP. It is distributed over the TTC in a dedicated message part of the L2 accept message and received by the detector readout electronics through the TTC Rx chips. When running without the ALICE Trigger system, the Event ID 1 field is set to zero and the Event ID 2 contains an incremental, unsigned binary number, to be reset at front-end electronics reset.

Block Attributes

The Block Attributes is an optional field that can be freely used by the detector groups to encode specific information.

Participating Sub-Detectors

The mask of participating detectors is a mandatory field. Its information is produced by the CTP only while handling software triggers. It is distributed over the TTC in a dedicated message part of the L2 accept message and received by the TTC Rx chips. The received values are copied as is in the Participating Sub-Detectors field.

Status & Error Bit

This is a mandatory field, to be loaded by the readout electronics under all running conditions. An error or status condition occurring before, during, or right after the front-end electronics readout is signalled by setting to one the corresponding bit(s) of this field. The assignment of the individual bits is described in Table 1.2.

Table 1.2: Status and error bits definitions.

Bit 12	Trigger overlap error: (L1 received while processing another L1)
Bit 13	Trigger missing error: (L1 received while no L0 received, or L2a or L2r received while no L1 received)
Bit 14	Data parity error
Bit 16	Control parity error (instruction and/or address)
Bit 17	Trigger information unavailable
Bit 18	Front-end electronics error
Bits 19 to 27	Reserved for future use

Mini-Event ID

The Mini-Event ID is the value of the bunch-crossing counter of the local TTC Rx chip at the time of the detector readout. It is a mandatory field. When running without the ALICE Trigger system, the Mini-Event ID field is set to zero.

A local event identification is also included in the common data format for cross-verification with the global event identification. This local event identification identifies the local value of the LHC clock at the time the detector has received the L1 trigger signal. It is based on the event identification reduced to the bunch crossing only, as counted by the TTC Rx chip [10]. The local bunch-crossing counters of all the TTC Rx chips of the experiment must be synchronous.

A key issue is to resynchronize them at regular intervals to ensure that this synchronism is maintained. The solution chosen is to use the mechanism foreseen by the TTC system. The local bunch-crossing counter in the TTC Rx chip can be automatically reset by a fast signal, synchronous with the LHC orbit. The LHC orbit signal is delivered by the TTCmi module [11]. This signal can then be sent over the TTC as a short-format broadcast signal. Proper usage and setting of the TTCvi module [12] guarantees that the TTC Rx chip receives this reset command by the end of the LHC extractor gap. The TTCvi includes four independently programmable timing signals. The bunch counter reset command uses the highest-priority programmable timing signal of the TTCvi.

Trigger classes (Low & High)

For physics triggers, the bits encoded in the Trigger classes' low and high fields are taken as is from the Trigger Level 2 accept message.

Region Of Interest (ROI) (Low & High)

The ROI data is distributed over the TTC system. The value—if available—should be stored in the ROI Low and ROI High fields. These fields are optional.

1.2.4 Central Trigger Processor readout and interaction-record data format

The CTP readout and the interaction-record data are generated by the CTP and transmitted to the DAQ via the DDL. The hardware and the communication procedure are standard—identical to the channels that transmit the sub-detector readout. The nature of the data, and the timing and rate of their generation, on the other hand, differ significantly from the sub-detector readout and are formatted by a customized data format. The CTP readout contributes to the event-building task. It is a redundant channel that carries exactly the same information broadcast to all the participating sub-detectors (L2a Message) at the time of L2a decision. It is used by the DAQ system to resolve error conditions.

The Interaction Record is an aid in pattern recognition. The generation of the record is continuous, rather than triggered by any CTP or DAQ action. Data do not interfere with any on-line operation—they only need to be archived for off-line use. Inside the DAQ, data from the CTP are formatted by the dedicated software to become compatible with the standard readout format used by the sub-detectors.

The CTP readout data format

For each L2a decision, the CTP transmits to the DAQ a data block containing the same information that is also broadcast to all the participating sub-detectors. The CTP readout block is shown in Fig. 1.3. The abbreviations used are summarized in Table 1.3.

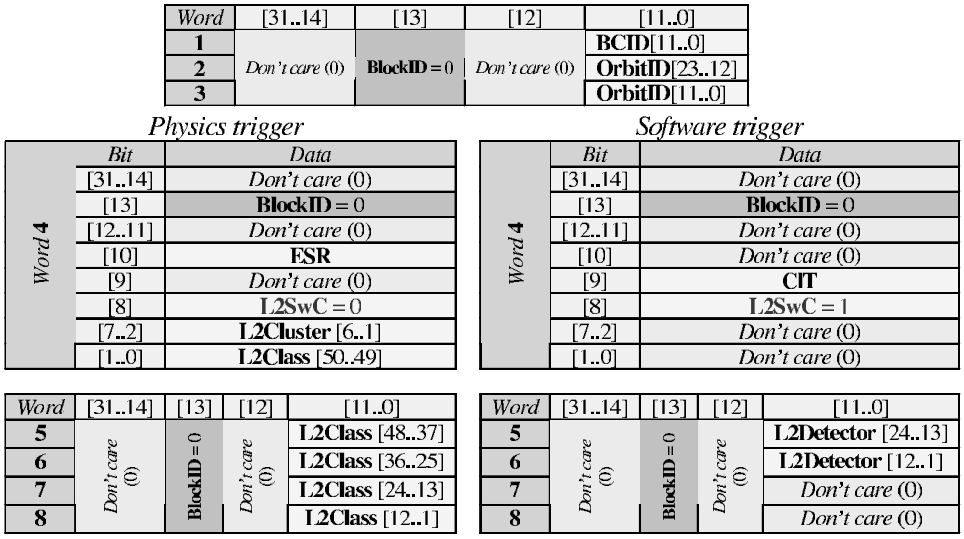

Figure 1.3: CTP readout data format.

Table 1.3: Abbreviations used in the CTP readout data format.

BlockID	Block Identifier bit: cleared (0) for the CTP readout asserted (1) in case of the Interaction Record
BCID[11...0]	Bunch-crossing number, part of the Event Identifier
OrbitID[23...0]	Orbit number, part of the Event Identifier
ESR	Enable Segmented readout flag (RoI option)
L2SwC	Software Class L2 trigger status: cleared for the physics trigger; asserted for the software trigger
L2Cluster[6...1]	Cluster [6...1] L2 trigger status flag
L2Class[50...1]	Class [50...1] L2 trigger status flag
CIT	Calibration Trigger flag
L2Detector[24...1]	Detector [24...1] L2 trigger status flag

The CTP sends data blocks of a constant length of seven words to the DAQ. The first three words always contain the Event Identifier (bunch-crossing and orbit number of the corresponding event). The remaining words (Word 4–8) carry different information in the case of the physics trigger (L2SwC = 0) and in the case of the software trigger (L2SwC = 1).

The interaction-record data format

For every interaction detected, the CTP transmits to the DAQ the bunch-crossing number and the type of interaction (peripheral or semi-central). These informations are packed together into interaction-record data block. There is at least one interaction-record data block for each LHC orbit, even if no interaction has been detected during the orbit. The data format is shown in Fig. 1.4. The abbreviations used are summarized in Table 1.4.

The first two words of the data block contain the number of the corresponding LHC orbit (24 bits). They are followed by a string of words containing the numbers (12 bits) of the bunch-crossing at which the interactions have been detected and the corresponding Interaction-Type (InT) descriptors. One Interaction Record is sent to the DAQ every 250 interactions. During a run, the DAQ receives, in sequential order, the Interaction Record data blocks for all the LHC orbits.

Word	[31..14]	[13]	[12]	[11..0]
1	*Don't care* (read 0)	**BlockID** = 1 (*Interaction Record*)	**Err**	**Orbit** number [23..12]
2			**Err**	**Orbit** number [11..0]
3			**InT**	**BC** number [11..0]
4			**InT**	**BC** number [11..0]
⋮			⋮	⋮
n			**InT**	**BC** number [11..0]
⋮			⋮	⋮
251			**InT**	**BC** number [11..0]
252			**InT**	**BC** number [11..0]
253			0	*Incomplete record* (hFFF)
254			0	**EOBTR** (h0B4)

Figure 1.4: Interaction Record data block.

Table 1.4: Abbreviations used in the interaction readout data format.

BlockID	Block Identifier bit: clear (0) for the CTP readout set (1) in case of the Interaction Record
BCID[11...0]	Bunch-crossing number, part of the Event Identifier
Err	Transmission Error flag
InT	Interaction Type flag: cleared (0) for peripheral and asserted (1) for semi-central interaction

1.2.5 High-Level Trigger readout

The HLT system transfers three kinds of information to the DAQ system:

- The HLT decision tells the DAQ whether an event has to be read out or not. It can take different values: a rejection, a complete readout, or a partial readout. The HLT decision is implemented as a refined list of sub-events that have to be read out. This is a refinement of the decision taken by the Trigger system.

- The data modified by the HLT data compression. Again, here this will be implemented as a refined list of subevents: replacing a set of sub-events by another one containing the same data but in a processed form.
- New data produced by the HLT such as Event Summary Data (ESD).

1.2.6 The DATE ROOT recorder

The requirements for the DATE recorder are the following:

- It is part of the DAQ framework for the dataflow and control aspects.
- It writes the physics data in ROOT format and formats the data into objects.
- It uses CASTOR as Mass Storage System (MSS) and is interfaced to the Grid for the file catalogue and the metadata repository.

The DATE recorder has therefore the following characteristics:

- The recorder process is started and stopped by the DATE Run Control in synchrony with the rest of the DAQ system.
- It gets the data as a set of pointer and size to the different subevents which constitute one event and writes all the data pertaining to the same event into the same file.
- The DATE recorder archives the raw events as one or several streams of C++ ROOT objects in TDS. The data files are then archived to the CASTOR MSS.
- The ROOT objects are formatted by a library included in the offline software. These objects are formatted to allow an easy access and an efficient processing by the ALICE Offline reconstruction software. The objects include a set of physics raw data, and a set of ESD.
- The DATE recorder is also interfaced to the Grid-based run catalogue database that contains information on the location of the raw events, the description of each recording unit, and some run-related information (trigger parameters and run conditions).

1.3 Computing data challenges

1.3.1 Introduction

Since 1998, the ALICE team and the CERN/IT division have jointly executed several Computing Data Challenges (CDCs): cadenced, large-scale, high-throughput, and distributed-computing exercises, whose goals are to prototype the ALICE Data Acquisition and computing systems, to test hardware and software components, and to accomplish an early integration of the overall ALICE computing infrastructure.

The first CDC took place in 1998. Six CDCs have been held so far. Year after year, existing components have been replaced with new versions and new elements have been introduced in the chain. The exercise started by using material already deployed at CERN such as Test-Beam Data-Acquisition systems, Fast-Ethernet, and the Gigabit-Ethernet based backbone and by exercising the system with dummy data.

As the exercise evolved in time and requirements, the focus has shifted towards a continuous early-prototyping effort and to the deployment of dedicated, highly-available fabrics, avoiding interference with other CERN activities whenever possible. The data used during the CDC have evolved into realistic physics data simulated with the ALICE simulation program.

Goals of the Computing Data Challenges

The main goals of the CDCs are the following:

- integrate and test the most recent version of all the components of the DAQ fabric and the DAQ software, its interface with the ALICE offline software, and the Mass Storage System (MSS);
- measure the performance of the complete setup over long periods.

The performance measurements are executed with simulated data pushed through the DATE software framework. DATE performs a coordinated injection of the simulated data, and the event building. The data are then formatted with the DATE recorder based on the ROOT I/O package and AliRoot. A data catalogue is also created during the CDC. This catalogue is based on the Grid software (AliEn). The Mass Storage System (MSS) used is CASTOR. During these tests, the following items are measured:

- scalability of the DAQ system to control and configure hundreds of computing nodes;
- sustained aggregate throughput of the data transfer and the event-building inside the DAQ for several hours;
- sustained aggregate throughput data of the whole chain including recording to the permanent data storage system for seven consecutive days;
- global amount of data being recorded to the permanent data storage system.

The latest available hardware and software version of the following components have been tested:

- network technologies: trunking, backbone switching, Gigabit Ethernet and 10-Gigabit Ethernet;
- commodity hardware: hosts, network interface cards, tape units, and tape robots;
- Several DATE versions have been used to simulate the behaviour and the output of an ALICE-like experiment;
- ALICE fabric monitoring software (AFFAIR) [1] (page 245) to assess the behaviour of the components of the DAQ and the interface to the MSS;
- ALICE Offline software: objectification of raw data, handling of event objects, recording and interfacing to the MSS;
- CASTOR—deployed on CPU servers, disk servers and tape servers—for MSS functions;
- operating system (Linux) with its kernel, system, user libraries, drivers, file systems (local and networked), network daemons (standard and custom-designed).

Data traffic in use

The data used during the CDCs are simulated physics raw data for most of the ALICE detectors. These data are produced with the ALICE simulation program AliRoot [13] before the beginning of the Data Challenge, then split into several sub-events and injected by several computers into the data acquisition fabric.

For the purposes of the CDC, data are created by reading special data files into the *readout* process memory space during the start-of-run phase and then events are created using these data, as directed by a fixed configuration sequence. The data flow from the LDCs is created using the Configurable LDC Emulator (COLE) package. By configuring the sequence of events, their data, and their characteristics, it is possible to create several different patterns of data traffic: the ALICE data traffic, the ALICE events, and the ALICE simulated events.

- The ALICE data traffic is characterized by all LDCs sending a realistic amount of data, similar to that which they are supposed to send under ALICE running condition. The LDCs, their Network Interface Cards (NIC), and up-links handle realistic data blocks while the GDCs and their associated network resources are less loaded as they are supposed to be in real-life conditions. On account of the number of available LDCs (40 instead of 200 in the final DAQ system), one fifth of the ALICE nominal capacity could be simulated so far. The GDCs therefore handle events five times smaller than the real ALICE events.

- The ALICE events pattern creates at the output of the GDCs events whose size is equivalent to that which we expect at the same level under normal ALICE running condition. In this scenario, the LDCs create and handle events five times bigger than in real life while the GDCs write as much data as they are expected to do during ALICE heavy-ion data-taking periods.

- Finally, the ALICE simulated events scenario introduces a set of simulated events in the data stream. In the present implementation, several events are available for the same trigger classes and this varies the data flow coming from the LDCs quite considerably. During CDC V, this scheme was adopted for selected types of events (central and semi-central) and for some of the ALICE detectors (TPC, SDD, SSD, and SPD) for which simulated data are available.

For the three scenarios described above, a fixed sequence of events and trigger types that follows a distribution similar to that which is expected within the ALICE experiment during heavy-ion runs is created. This sequence implies the participation of selected detectors according to the trigger classes associated with each event and partial read-out is also applied for dielectron events.

1.3.2 Performances and results

Results of CDC III and CDC IV can be found in Refs. [14, 15]. Here we report in detail the objectives and the results of the last two CDCs, CDC V, held in 2003 and CDC VI that took place at the end of 2004 and beginning of 2005.

Figure 1.5 shows the planning of the ALICE Data Challenges as a function of the targeted data rates through the Data-Acquisition system (DAQ bandwidth) and recorded by the Mass Storage (Mass Storage bandwidth). The expected available tape bandwidth in the Tier 0 is also shown.

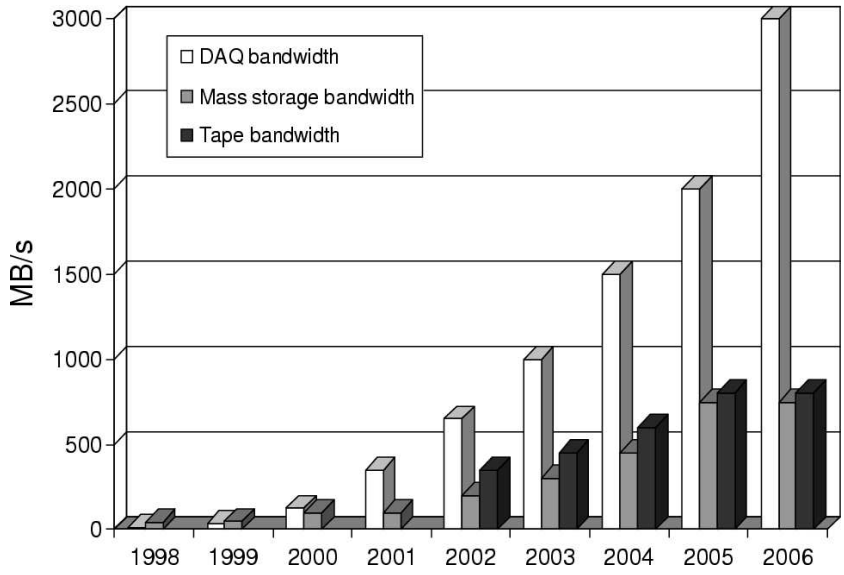

Figure 1.5: ALICE Data Challenge bandwidth planning.

As indicated in Fig. 1.5, the target is to increase the data rates progressively until something as close as possible to the final ALICE requirements—in agreement with the available hardware resources allocated to the exercise—is met. This will happen no later than one year before LHC startup.

As an example of the successfully reached milestones, Fig. 1.6 shows the sustained throughput achieved during CDC IV in 2002. The final results were a peak rate of 310 MB/s, a sustained average data rate produced by the DAQ System of 280 MB/s, and 180 TB moved onto the PDS over a period of seven consecutive days. The CDC IV was also a large-scale test combining 32 and 64 bits machines.

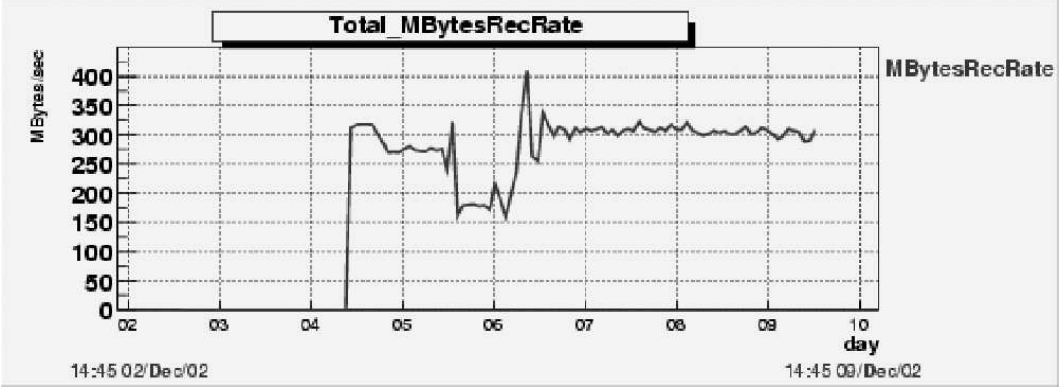

Figure 1.6: Sustained throughput milestone monitoring.

During CDC VI in 2005, higher sustained throughputs have been achieved with 2.5 GB/s of DAQ bandwidth (see Fig. 1.7) and an average of 450 MB/s to PDS with the RFIO format (see Fig. 1.8).

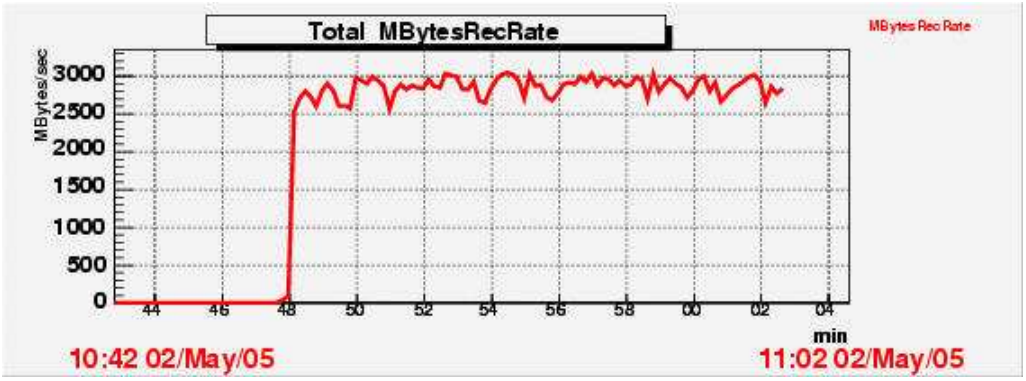

Figure 1.7: Sustained throughput of the DAQ system.

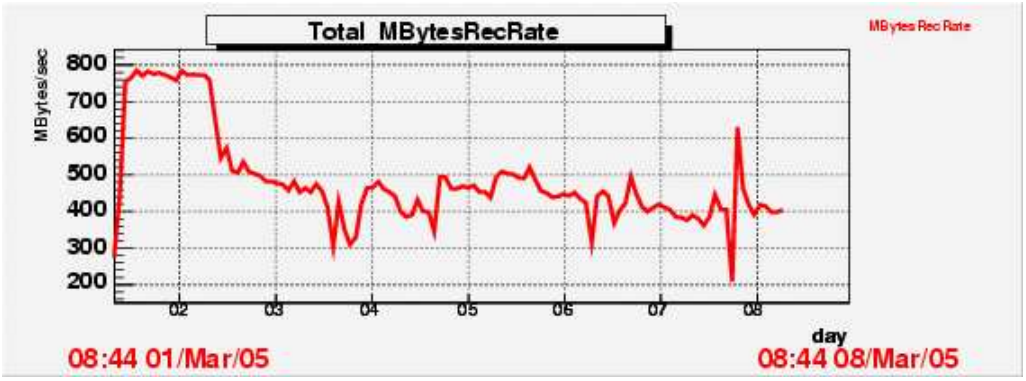

Figure 1.8: Sustained throughput to CASTOR.

Another important task during the CDCs is the online physics monitoring. The online physics monitoring is a software module developed within the AliRoot program framework. It runs fast physics event-reconstruction based completely on the available HLT software. The two main goals of the monitoring are to check the consistency of the raw data coming from the DAQ system and to test the performance of the HLT algorithms in an environment as realistic as possible. The monitoring code is run in two basic modes: online and quasi-online. In the online mode, the simulated raw data, after the event building, is processed and the HLT event summary data is written together with the raw data itself. The preliminary tests show very good time performance of the monitoring—ranging between 5 s and 10 s for processing and reconstruction of the TPC and the ITS raw data for central Pb–Pb events. The tracking efficiency depends on the HLT algorithms used and in general is about 5–10% lower than that of the standard ALICE offline reconstruction. The second operation mode of the online physics monitoring—the quasi-online one—makes use of the same HLT code, but in addition also runs some parts of the offline reconstruction chain. In this mode, the monitoring takes the files with raw data events already stored and registered on the Grid, processes them,and fills series of monitoring histograms. Owing to the flexibility of the developed framework, the user can be provided with fast feedback related to the quality of the raw data as well as to the physics performance of an individual sub-detector or the entire ALICE setup.

1.3.3 Future plans

The goals of the next Data Challenge (CDC VII) include the following points:

- realize the test in real conditions with the DAQ setup at the experimental area (LHC point 2) and the Tier 0 equipment in the computing centre;
- record the data in ROOT format;
- increase the number of pieces of equipment to be read;
- improve the profile of the data traffic including realistic trigger;
- add the contribution from hardware-driven data sources (such as the DDL data pattern generator);
- test new network technologies and topologies;
- further develop and optimize the online physics monitoring framework;
- test the online physics monitoring software for other ALICE sub-detectors;
- achieve higher performance.

2 Overview of the computing framework

The objective of the ALICE computing framework is twofold: the simulation of the primary pp and heavy-ion interactions and of the resulting detector response; and the reconstruction and analysis of the data coming from simulated and real interactions.

When building the ALICE detector, the optimization of the hardware design and the preparation of the code and computing infrastructure require a reliable simulation and reconstruction chain implemented by a distributed computing framework. Since few years, ALICE has been developing its computing framework called AliRoot [1]. This has been used to perform simulation studies for the Technical Design Reports of all ALICE sub-detectors, in order to optimize their design. It has also been used for the studies of the ALICE Physics Performance Report to assess the physics capabilities of the ALICE detector and for the Computing and Physics Data Challenges. A representation of the ALICE detector geometry within the AliRoot simulation framework is illustrated in Colour Figure I. This chapter describes the development and present structure of the AliRoot framework.

2.1 The development of AliRoot

The development of the ALICE computing framework started in 1998. At that time, the size and complexity of the LHC computing environment had already been recognized. Since the early 1990s it had been clear that the new generation of programs would be based on Object-Oriented (OO) techniques [2]. A number of projects were started, to replace the FORTRAN CERNLIB [3], including PAW [4] and GEANT 3 [5] with OO products.

In ALICE, the computing team developing the framework and the physicists developing physics software and algorithms are in a single group. This organization has proved effective in improving communication, and in encouraging the software developers to constantly provide working tools while progressing towards the final computing system. Thanks to this close collaboration, the ALICE physics community supported a rapid and complete conversion to OO/C++, under the condition that a working environment at least as good as GEANT 3, CERNLIB and PAW could be provided quickly.

After a short but intense evaluation period, the ALICE computing team concluded that one such framework existed, namely the ROOT system [6], which is now the de facto standard of HEP software and a major foundation of the software of the LHC Computing Grid (LCG) [7] project. The decision to adopt the ROOT framework was therefore taken and the development of the ALICE computing framework, AliRoot, started immediately, making full use of all the ROOT potential.

This process has resulted in a tightly knit computing team with one single line of development. The move to OO was completed successfully resulting in one single framework, entirely written in C++ and soon adopted by the ALICE users. The detector Technical Design Reports and the Physics Performance Report were written using simulation studies performed with this framework, while the system was in continuous development. This choice allowed the ALICE developers to address both the immediate needs and the long-term goals of the experiment with the same framework as it evolved into the final one, with the support and participation of all ALICE users.

2.2 ROOT framework

The ROOT framework[1], schematically shown in Fig. 2.1, provides a set of interacting classes and an environment for the development of software packages for event generation, detector simulation, event

[1] In HEP, a framework is a set of software tools that enables data processing. For example CERNLIB was a toolkit to build a framework. PAW was the first example of integration of tools into a coherent ensemble specifically dedicated to data analysis. ROOT is a new-generation, Object-Oriented framework for large-scale data-handling applications.

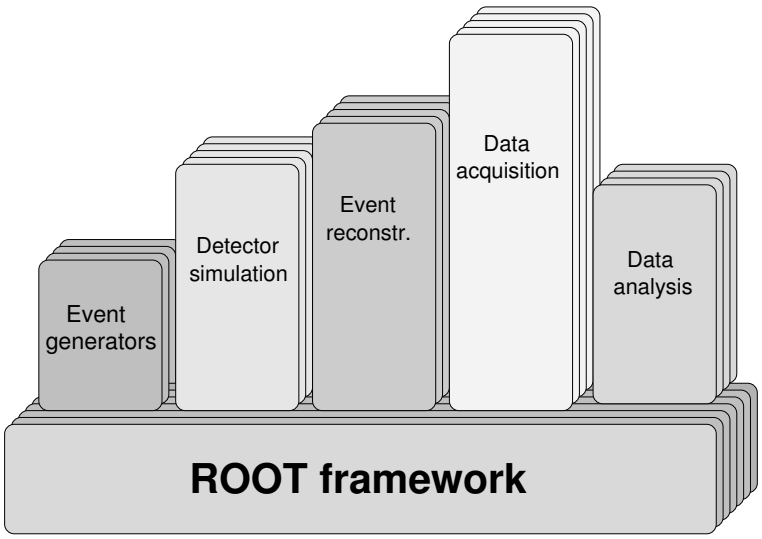

Figure 2.1: The ROOT framework and its application to HEP.

reconstruction, data acquisition, and a complete data analysis framework including all PAW features. An essential component of ROOT is the I/O subsystem that allows one to store and retrieve C++ objects and is optimized for efficient access to very large quantities of data.

ROOT has evolved over the years to become a mature and complete framework, embodying a very large spectrum of features. It is outside the scope of this document to provide a full description of ROOT. In the following paragraphs we shall limit ourselves to outlining the major features that are relevant for the ALICE computing framework.

ROOT is written in C++ and offers integrated I/O with class schema evolution, an efficient hierarchical object store with a complete set of object containers, a C++ interpreter allowing one to use C++ as scripting language, advanced statistical analysis tools (multidimensional histogramming, several commonly used mathematical functions, random number generators, multi-parametric fit, minimization procedures, cluster finding algorithms etc.), hyperized documentation tools, geometrical modelling, and advanced visualization tools. The user interacts with ROOT via a graphical user interface, the command line or script files in C++, which can be either interpreted or dynamically compiled and linked.

ROOT presents a coherent and well integrated set of classes that easily inter-operate via an object bus provided by the interpreter dictionaries (these provide extensive Run Time Type Information, RTTI, of each object active in the system). This makes ROOT an ideal basic infrastructure on which an experiment's complete data handling chain can be built: from DAQ, using the client/server and shared memory classes, to database, distributed analysis, thanks to the PROOF facility, and data presentation.

The Parallel ROOT Facility [8], PROOF, makes use of the inherent parallelism in event data and implements an architecture that optimizes I/O and CPU utilization in heterogeneous clusters with distributed storage. Being part of the ROOT framework, PROOF inherits the benefits of a well-performing object storage system and a wealth of statistical and visualization tools. Queries are automatically parallelized and balanced by the system which implements a simple but very effective master–worker model. The results of the queries are joined together by the system and presented to the users.

The backbone of the ROOT architecture is a layered class hierarchy organized in a single-rooted class library where classes inherit from a common base class `TObject`. This has proved to be well suited for our needs (and indeed for almost all successful class libraries: Java, Smalltalk, MFC, etc.). A ROOT-based program links explicitly with a few core libraries, and at run-time it loads dynamically additional libraries as needed, possibly via string activated class initializations (plugins).

One of the key components of ROOT is the CINT C/C++ interpreter. The ROOT system embeds CINT in order to be able to execute C++ scripts and C++ command line input. CINT also provides

ROOT with extensive RTTI capabilities covering essentially the totality of C++. The advantage of a C/C++ interpreter is that it allows for fast prototyping since it eliminates the typical time-consuming edit/compile/link cycle. Scripts can be compiled on-the-fly from the ROOT prompt with a standard C/C++ compiler for full machine performance.

ROOT offers a number of important elements that have been exploited in AliRoot as the basis for the successful migration of the users to OO/C++ programming.

The ROOT system can be seamlessly extended with user classes that become effectively part of the system. The corresponding libraries are loaded dynamically and the user classes share the same services of the native ROOT classes, including object browsing, I/O, dictionary and so on.

ROOT can be driven by scripts having access to the full functionality of the classes. For production, it eliminates the need for configuration files in special formats, since the parameters can be entered via the setters of the classes to be initialized. This is also particularly effective for fast prototyping. Developers can implement code working with a script within the same ROOT interactive session. When the code is validated, it can be compiled and made available via shared libraries, and the development can restart via scripts. This has been observed to lower the threshold considerably for new C++ users and has been one of the major enabling factors in the migration of users to the OO/C++ environment.

The ROOT system has been ported to all known Unix variants (including many different C++ compilers), to Windows from 9x to XP, and to Mac OS X.

ROOT is widely used in particle and nuclear physics, notably in all major US labs (FermiLab, Brookhaven, SLAC), and most European labs (CERN, DESY, GSI). Although initially developed in the context of particle and nuclear physics, it can be equally well used in other fields where large amounts of data need to be processed, such as astronomy, biology, genetics, finance, insurance, pharmaceutical, etc.

ROOT is the foundation of the LCG software, providing persistency for LHC data and data analysis capabilities.

The ROOT system has recently been interfaced with emerging Grid middleware in general. A first instantiation of this interface was with the ALICE-developed AliEn system [9]. In conjunction with the PROOF system, this has allowed the realization and demonstration of a distributed parallel computing system for large-scale production and analysis.

This interface is being extended to other middleware systems as these become available.

2.3 AliRoot framework

2.3.1 The function of AliRoot

The funrcionality of the AliRoot framework is shown schematically in Fig. 2.2. Simulated data are generated via Monte Carlo event generators. The generated tracks are then transported through the detector via detector simulation packages such as GEANT 3, FLUKA [10] and GEANT 4 [11]. These packages generate a detailed energy deposition in the detector, which is usually called a ‘hit’, with a terminology inherited from GEANT 3. Hits are then transformed into an ideal detector response, which is then transformed into the real detector response, taking into account the electronic manipulation of the signals, including digitization. As explained in Chapter 4 on simulation, the need to ‘superimpose’ different simulated events has led us to develop an intermediate output format called ‘summable digits’. These are high-resolution, zero-threshold digits, which can be summed when different simulated events are superimposed. Digits are then transformed into the format that will be output by the electronics of the detectors, called ‘raw’ format. From here on the processing of real or simulated data is indistinguishable.

The data produced by the event generators contain the full information about the generated particles: Particle Identification (PID) and momentum. As these events are processed via the simulation chain, the information is disintegrated and reduced to that generated by particles when crossing a detector. The reconstruction algorithms reconstruct the information about the particle trajectory and identity from the information contained in the raw data. In order to evaluate the software and detector performance,

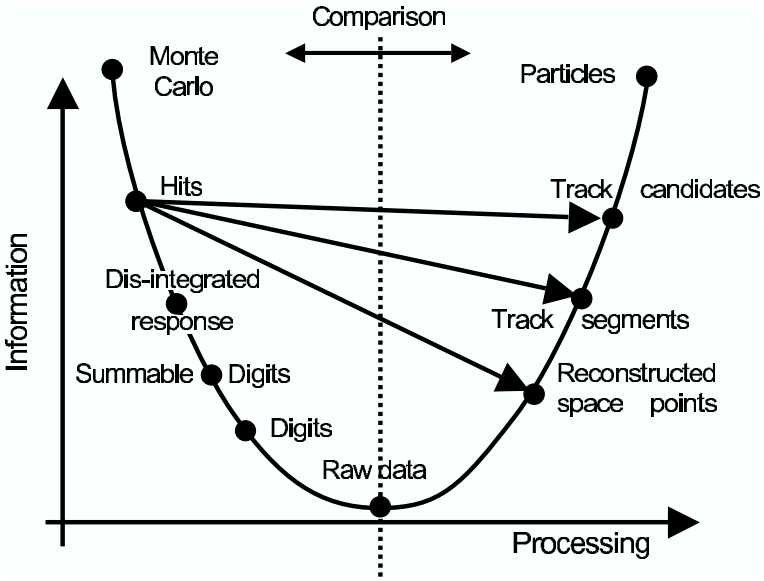

Figure 2.2: Data processing framework.

simulated events are processed through the whole cycle and finally the reconstructed information about particles is compared with the information taken directly from the Monte Carlo generation.

Fast simulations are 'shortcuts' in the whole chain, as indicated by the arrows in the figure. These increase the speed of the simulation at the expense of the details of the result, and are used for special studies. The AliRoot framework implements several fast simulation algorithms.

The user can intervene in this framework-driven cycle to implement private code provided it respects the interfaces exposed by the framework. I/O and user interfaces are part of the framework, as are data visualization and analysis tools and all procedures and services of general interest. The scope of the framework evolves with time following the needs and understanding of the users.

2.3.2 Principles of AliRoot design

The basic principles that have guided us in the design of the AliRoot framework are re-usability and modularity, with the objective of minimizing the amount of unused or rewritten user code and maximizing the participation of the physicists in the development of the code.

Modularity allows the replacement of parts of the system with minimal or no impact on the rest. Not every part of the system is expected to be replaced and modularity is targeted at those elements that could potentially be changed. There are elements that we do not plan to change, but rather to develop in collaboration with their authors. Whenever an element has to be modular in the sense above, we define an abstract interface to it. Some major examples are:

- Different transport Monte Carlo's can be used without change in the scoring or geometry description codes for the different sub-detectors. This is further described in detail in Chapter 4. This is realized via a set of virtual interfaces that are part of the ROOT system, called Virtual Monte Carlo.
- Different event generators can be used and even combined via a single abstract interface.
- Different reconstruction algorithms for each sub-detector can be used with no effect on the rest of the code. This includes also the algorithms being developed for HLT, to allow their comparison with the offline ones. Again this is implemented via abstract interfaces.

The codes from the different detectors are independent so that different detector groups can work concurrently on the system while minimizing interference.

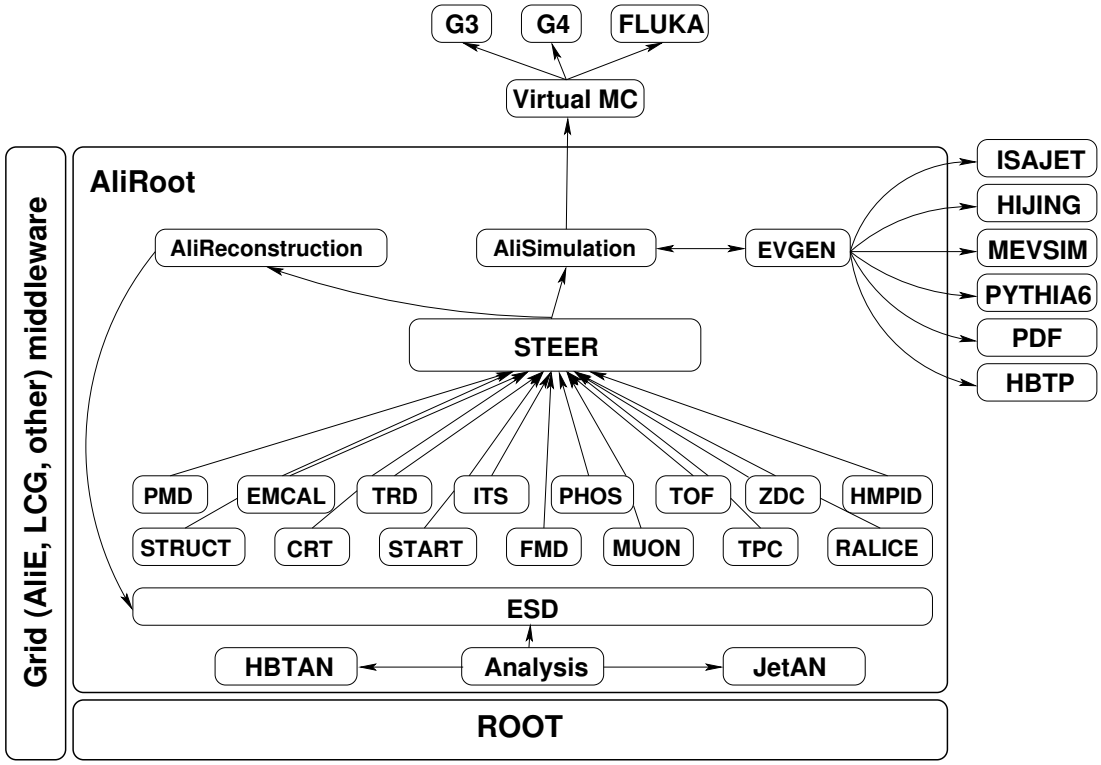

Figure 2.3: Schematic view of the AliRoot framework.

Re-usability is the protection of the investment made by the physicist programmers of ALICE. The code contains a large amount of scientific knowledge and experience and is thus a precious resource. We preserve this investment by designing a modular system in the sense above and by making sure that we maintain the maximum amount of backward compatibility while evolving our system.

The AliRoot framework is schematically shown in Fig. 2.3. The central module is STEER and it provides several common functions such as:

- steering of program execution for simulation, reconstruction and analysis;
- general run management, creation and destruction of data structures, initialization and termination of program phases;
- base classes for simulation, event generation, reconstruction, detector elements.

The sub-detectors are independent modules that contain the specific code for simulation and reconstruction while the analysis code is progressively added. Detector response simulation can be performed via different transport codes via the Virtual Monte Carlo mechanism [12].

The same technique is used to access different event generators. The event generators are accessed via a virtual interface that allows the loading of different generators at run time. Most of the generators are implemented in FORTRAN but the combination of dynamically loadable libraries and C++ 'wrapper' classes makes this completely transparent to the users.

2.3.3 Data structure design

Object-Oriented design is based on data encapsulation. Data processing in HEP is based on successive passes on data that is subsequently transformed and analysed. It has been noted that in this case data encapsulation may lead to tangled relationships between classes, introducing unwanted dependencies

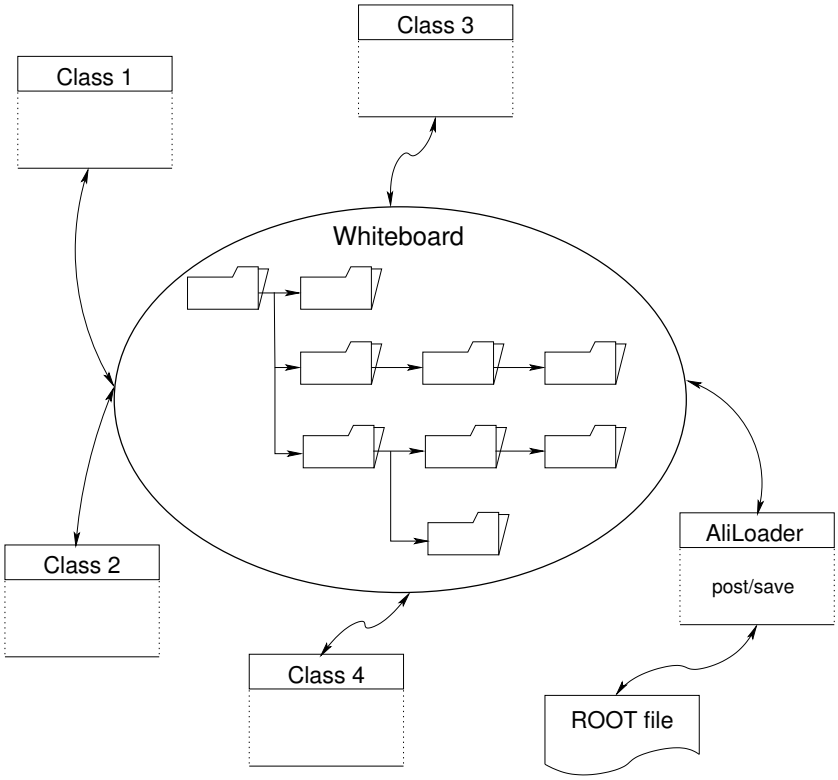

Figure 2.4: AliRoot whiteboard.

and making the design difficult to evolve. To avoid this problem and still maintain the benefits of Object-Oriented design, we have decided to exploit the ROOT folder facility to create a data 'whiteboard' as shown in Figure 2.4.

The main idea of the data whiteboard is that I/O is delegated to a set of service classes that read data from the files and 'post' them to a set of ROOT folders with the same semantics as a Unix file system. The same classes perform the opposite operation and 'unload' the data into a file. This has had the double advantage of concentrating I/O operations in a single set of classes, and simplifying data dependencies between modules, which therefore remain independent of each other. Figure 2.5 presents the AliRoot folder structure for the ESD.

2.3.4 Offline detector alignment and calibration model

The ALICE offline calibration and alignment framework will provide the infrastructure for the storage and the access to the experiment condition data. These include most non-event data and notably the calibration and alignment information used during reconstruction and analysis. Its design is primarily driven by the necessity for a seamless access to a coherent set of calibration and alignment information in a model where data and processing are distributed worldwide over many independent computing centres.

The condition data are contained in ROOT objects stored into read-only ROOT files and registered in the ALICE file catalogue. Evolution will be handled through versioning, thus avoiding the necessity to develop a complicated internal synchronization schema. The file catalogue will associate the metadata describing the condition information with the logical files where the information is stored. These tags will be the primary method for file access during the production and analysis phase.

The usage of the ROOT I/O object store capabilities together with the navigational, metadata and distributed access capabilities of the ALICE Grid file catalogue avoids the development of a more complicated but functionally equivalent structure based on distributed relational database systems.

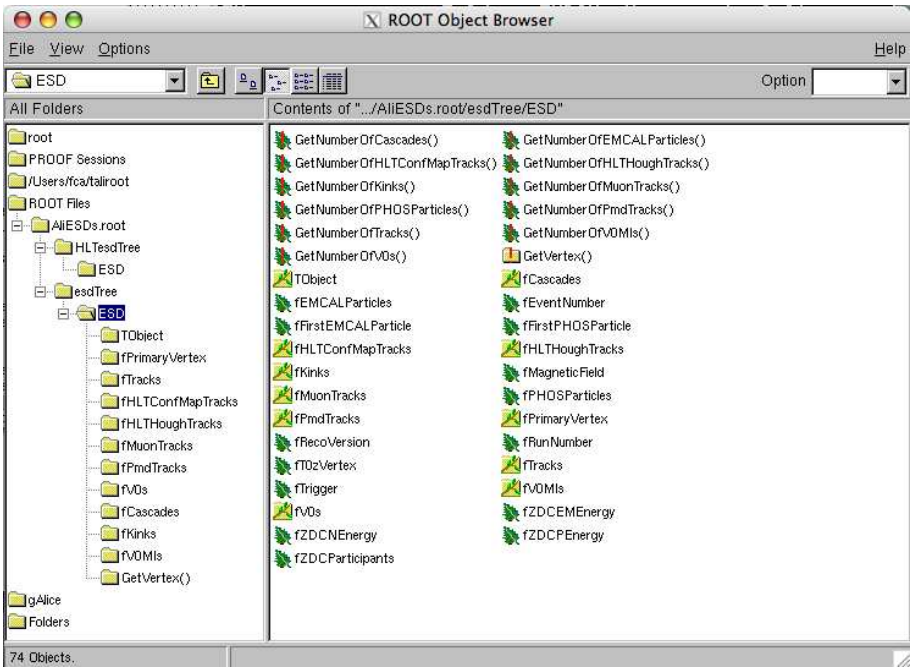

Figure 2.5: Example of AliRoot ESD folders.

The framework provides a set of classes to access and manipulate the condition objects. These can be stored in a distributed, Grid-enabled environment or copied onto the local disk of a machine potentially disconnected from the network. In this case the metadata will be encoded in the file name. The classes are designed so that, once the access method is specified, no other change in the code is necessary.

The source data for condition information, both static and dynamic will be stored in on-line and archive databases, which are populated during the detector construction and integration phase and the data taking periods. These databases include:

- Detector Construction Database (DCDB): Used by individual sub-detector groups during the production and integration phase and containing static information on detector elements performance, localization and identification.

- Experiment Control System (ECS) database: Contains information on the active sub-detector partition during data taking and the function of this partition.

- Data Acquisition (DAQ) database: A repository for data acquisition related parameters and for resource assignments to data acquisition tasks: current and stored configurations, current and stored run parameters.

- Trigger database: Contains the ALICE trigger classes (including the input to the Central Trigger Processor), and the definition of the trigger masks.

- Detector Control System (DCS) databases: Configuration DB, containing the configuration parameters for systems and devices (modules and channels), and the front-end configuration (busses and thresholds); Archive DB containing the monitored detectors and device parameters.

- High-Level Trigger (HLT) database: A TAG/ESD database containing HLT information relevant for physics studies and offline event selection.

- LHC Machine database: Machine status and beam parameters.

The alignment and calibration framework will collect the information from the various sources and make it accessible for offline distributed processing, either via periodic polling or asynchronous access. The information will be stored in ROOT files published in the ALICE Grid catalogue and annotated with the proper condition tags in the form of file metadata.

Most of the data flow is uni-directional toward the offline with the exception of the HLT system, which requires access to the condition data but may also generate them.

The relation of the alignment and calibration procedures and the various sources of condition data are presented in Fig. 2.6.

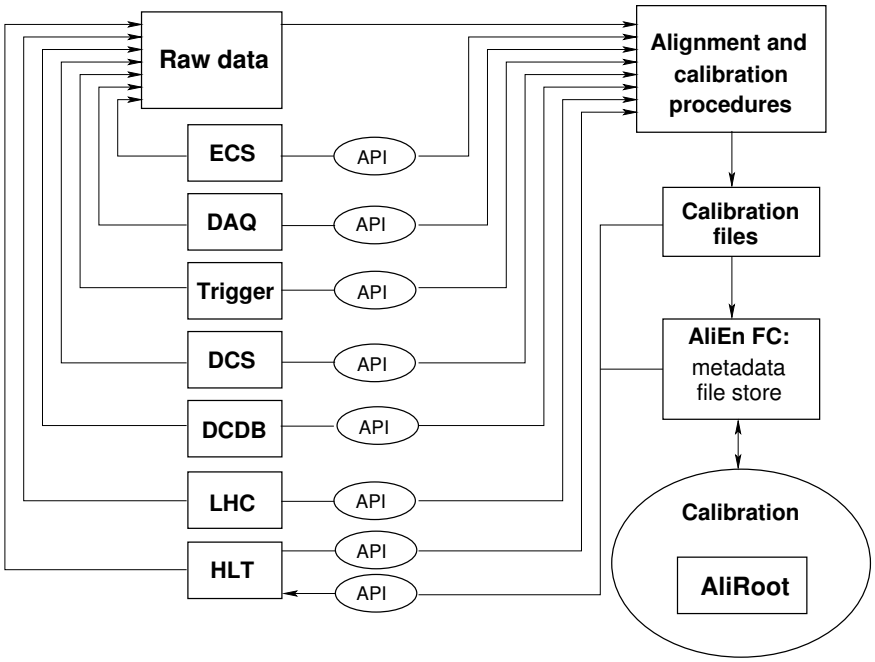

Figure 2.6: Schematic view of the relations of the calibration and alignment procedures to the condition data sources and the AliRoot and AliEn frameworks.

The update and access frequency to the condition objects will be once per run. The objects, however, can contain more finely-grained information depending on time, event number or other variables, for example in the form of a histogram. Each sub-detector will have its own condition storage partition, with possibly a different granularity and condition parametrization within the objects. There will be partitions containing information common for all ALICE sub-detectors, for example magnetic field maps and the LHC machine information. This design has been developed and verified by collecting and analyzing an extensive set of user requirements and use cases.

The sub-detectors and global alignment framework will be based on the ROOT geometrical modeller package [13]. The common alignment framework will provide facilities for the retrieval of the right alignment files and for the 'correction' of the position of the detectr elements with the alignment data. The calibration algorithms are intrinsically detector-specific and therefore their development is under the responsibility of the sub-detector groups.

At the time of publication of this document, work is going on in parallel on the development of the framework access classes, the API for gathering information from the various sources of condition data and detector-specific alignment and calibration code. These will be tested during the distributed Physics Data Challenges in 2005 and 2006 and will be ready for the start of data taking in April 2007.

2.3.5 Tag database

The ALICE experiment is expected to produce many petabytes of data per year. Most user analyses are conducted on subsets of events. Typically, the events are organized into files stored on mass storage systems. The performance of analysis jobs strongly depends on file management functions such as finding what files contain the wanted events, locating the files, transferring the files to a convenient location for analysis and removing the files afterwards, or moving the analysis task to the file location.

ALICE is implementing a system that will reduce the time needed to perform an analysis by providing to each analysis code only the events of interest as defined by the users themselves. This task will be performed first of all by imposing event selection criteria within the analysis code and then by interacting with software that is designed to provide a file-transparent event access for analysis programs. The latter is derived from a product developed and used by the STAR [14] Collaboration.

This software, called Grid Collector (GC) [15], is designed to provide file-transparent event access for analysis programs. This software allows users to specify their requests for events as sets of conditions on physically meaningful attributes, such as triggers, production versions and other tags. The GC resolves these conditions into a list of relevant events and a list of files containing the events. It is able to locate the files and transfer the files as needed. Its main components are the Query Interpreter and the Event Iterator.

- A component called Bitmap Index [16] takes as input the values of selected attributes and generates indices poiting to the events.
- The Query Iterator takes the selection criteria provided by the users and translates them with the help of the produced indices into a list of files and events in these files that satisfy the selection.
- The Event Iterator interfaces the analysis framework to the GC. It retrieves the selected events and and passes them to the analysis code.

The current prototype of the TAG database stores information about several stages of data processing such as: LHC machine, run and detector information and a set of event features selected by the Physics Working Groups for fast selection of an event sample from the full data set. All information will be stored in ROOT files. Two scenarios are considered for their creation:

- 'On-the-fly' creation. The TAG files are created by the reconstruction code and registered in the ALICE file catalogue together with the ESDs.
- 'Post-processing'. After the produced ESDs are registered, a process will loop over all the registered files of each run in order to create the tag ROOT files and register them in the ALICE file catalogue.

We will use the GC Bitmap Index to produce the indices for every attribute stored in these TAG files. The user uses the GC's Query Interpreter to have fast and efficient access to the events of interest via a set of selection criteria on TAG attributes.

2.3.6 LHC common software

The LHC Computing Grid (LCG) was launched in March 2002 following the recommendations of the LHC Computing Review [17]. Its objective is to provide the LHC experiments with the necessary computing infrastructure to analyse their data. This should be achieved via the realization of common projects in the field of application software and Grid computing, deployed on a common distributed infrastructure. The project is divided into two phases, one phase of preparation and prototyping (2002–2005), and one of commissioning and exploitation of the computing infrastructure (2006–2008).

An overview Software and Computing Committee (SC2) has organized Requirement Technical Assessment Groups (RTAGs) defining requirements. These are received by the Project Execution Board that organizes the project activity in several workpackages.

ALICE has been very active in the setting up of this structure and ALICE members have served as chairs for several RTAGs. ALICE has elaborated a complete solution for handling the data, and intends to continue developing it while collaborating fully with the other experiments in the framework of the LCG project. ALICE welcomes sharing its software with the other experiments.

In the application area, the LCG project has decided to make ROOT one of the central elements of its development and ALICE fully supports this decision. ALICE sees its role in the LCG project as one of close collaboration with the ROOT team in developing base technologies (e.g. geometrical modeller, Virtual Monte Carlo) which are included directly in ROOT and made available to the other LHC experiments.

In the Grid technology area ALICE, succesfully deployed and used its AliEn Grid framework. We are currently working in collaboration with LCG and the other experiments [18] to maximise the use of the common Grid infrastructure provided by LCG, interfacing the ALICE-specific AliEn services to it.

2.4 Software Development Environment

The ALICE software community is composed of small groups of two–three people who are geographically dispersed and who do not respond hierarchically to the Computing Coordinator. This affects not only the organization of the computing project but also the software development process itself as well as the tools that have to be employed to aid this development.

This situation is not specific to ALICE. It is similar to the other modern HEP experiments. However, this is a novel situation for HEP. In the previous generation of experiments, during the LEP era, although physicists from different institutions developed the software, they did a large share of the work while at CERN. The size of modern collaborations and software teams makes this model impractical. The experiments' computing programs have to be developed by truly distributed teams whose members meet very infrequently.

To cope with this situation, ALICE has elaborated a software process inspired by the most recent Software Engineering trends, in particular by the extreme programming principle [19]. In a nutshell, traditional software engineering aims at reducing the occurrence of change via a complete specification of the requirements and a detailed design before the start of development. After an attentive analysis of the conditions of software development in ALICE, we have concluded that this strategy cannot succeed. Therefore, we have adopted more flexible development methods and principles, which are outlined below:

- **Requirements are necessary to develop the code**. ALICE users express their requirements continuously with the feedback they provide. The computing core team redirects its priorities according to the user feedback. This way, we make sure that we are working on the highest priority item at every moment, thus maximizing the development efficiency. The developers meet at CERN three times per year during so-called 'offline weeks' that play a major role in reviewing the current state of the project, collecting user requirements and feedback, and planning of future activities.

- **Design is good for evolving the code**. AliRoot code is redesigned continuously since the feedback received from the users is folded directly into the design activity of the core computing team. At any given time there is a short-term plan to the next development phase, instead of the traditional long-term static objective.

- **Testing is important for quality and robustness**. While component testing is the responsibility of the groups providing modules to AliRoot, full integration testing is done nightly to make sure that the code is always functional. At present, the tests are concentrated on the main steps in

the simulation: event generation and transport, hits, detector response, summable digits, event merging, and digitization. In addition, the reconstruction is carried on including clusterization, reconstruction of tracks, vertices, V^0 and cascades, particle identification and creation of ESD. The results of the tests are reported on the Web and are publicly accessible [20].

- **Integration is needed to ensure coherence**. The AliRoot code is collectively owned with a single code base handled via the CVS [21] concurrent development tool, where all developers store their code. The same CVS repository is used by the HLT project to store their algorithms, which are functionally integrated into the offline code. The AliRoot CVS contains both the production version and the latest (development) version. For every module, one or two developers can perform updates. The different modules are largely independent and therefore, even if a faulty modification is stored, the other modules still work and the global activity is not stopped. Having all the code in a single repository allows for a global overview to be easily carried out [22]. A hyperized code version is also extracted using the ROOT documentation tools [23].

- **Discussions are valuable for users and developers**. Discussion lists are commonplace in all computing projects. In ALICE, the discussion list is used to express requirements and evaluate design. Frequently, important design decisions are initiated by an e-mail discussion thread, in which all interested ALICE users can participate. Following such discussion, a redefinition of planning and sharing of work can also occur, thus adding flexibility to the project and allowing it to address both short-term and long-term needs.

The above development strategy is very close to the one used by successful Open-Source projects such as Linux, GNU, ROOT, KDE, GSL and so on, making it easy for us to interact with them.

2.5 Maintenance and distribution of AliRoot

The ALICE code is composed of one single OO framework implemented in C++ and with a multitude of users and developers participating in the design and implementation. Distributed development of such a large and rapidly evolving code is a challenging task which requires proper organization and tools.

The development model chosen by ALICE implies that any design becomes quickly obsolete since the development is driven by the most urgent needs of the user community. In this environment it becomes of particular importance to avoid the risk of an anarchic code development with the introduction of well-known design errors (see for instance [24]). To keep the code development under control, we hold regular code and design reviews and we profit from advanced software engineering tools developed in collaboration with computer science experts, see Section 2.5.1

For this model to work, a specific release policy has been elaborated during the first two years of existence of AliRoot. It is based on the principle of fixed release dates with flexible scope. Given that the priority list is dynamically rearranged, it is difficult to define the scope of a given release in advance. Instead, we decided to schedule the release cycle in advance. The current branch of the CVS repository is 'tagged' every week with one major release every six months or before a major production. The production branch of the CVS repository is tagged only for new check-ins due to bug fixes.

When a release is approaching, the date is kept fixed, and the scope of what is to be included is tailored. Large developments are moved to the current branch and only developments that can be completed within the time remaining are included. The flexible priority scheduling ensures that if a feature is urgent, enough resources are devoted to make it ready in the shortest possible time. As soon as it is ready, it is made available via a CVS tag on the active branch.

Special software tools are used to improve and verify the quality of the code. We rely on the ROOT memory checker [25] for fast detection of memory leaks. Extensive searching for runtime errors is done using the Valgrind tool [26]. The VTune profiling tool [27] is extremely helpful in the optimization of the code.

The installation process of AliRoot is specifically tailored to be efficient and reliable. AliRoot is one single package that links only to ROOT. Thanks to this minimal dependency, the installation does not require configuration management tools. A special attempt has been made to be independent from the specific version of the operating system and compiler. To ensure easy portability to any future platform, one of our coding rules states that the code must compile without warning on all supported platforms. At this time, these are Linux IA32 and IA64 architectures, DEC-Unix with True64, Solaris, and Mac OS X.

2.5.1 Code development tools

Very early in the development of AliRoot we decided to acquire tools to help us with code development and maintenance via a collaboration with computer science experts. We therefore established a collaboration with the IRST [28–30] at Trento, Italy, who were interested in developing state-of-the-art software engineering tools and to test them in production on a large code.

Distributed and collective code development requires a high degree of uniformity of the code. Developers come and go, and the code and its structure has to be readable and easily understandable. We realized this very early on and therefore we decided to adopt a small set of coding and programming rules [31]. It was soon clear that only an automatic tool could verify the compliance of the code with these rules. This tool has been developed in collaboration with IRST and is currently used to check the code for compliance. A table of the violations in all modules is published on the Web [32] every day.

Object-Oriented code design and development is best done using formal representation of the code structure, such as the one provided by the Unified Modelling Language (UML [33]). The problem with UML is that it is difficult to maintain consistency between the code design and the actual code implemented. To alleviate this problem, associated with the code checker there is a reverse-engineering tool that produces UML diagrams for all the AliRoot modules. It is run together with the nightly builds so that an up-to-date formal representation of the code design is always available. The results are published on the ALICE offline Web page [34]. This tool is essential in providing a constantly up-to-date design of the AliRoot code, which is used in code design discussions.

A new project has been established with IRST for the period 2004–2007. The main objectives are to produce a new set of software engineering tools:

- **Automated test case generation**. Automated test case generation is used for coverage testing. The implementation is based on genetic algorithms. All the test cases are considered as individuals of a population with chromosome-encoded test input values. Test cases are evolved by means of mutation (e.g., change value) and crossover (e.g., swap input value tails). The fittest individuals are the test cases that get closest to the target of test execution (for example, covering a given branch).

- **Introduction of Aspect-Oriented programming**. This paradigm permits the execution flow to be intercepted by an aspect in a well-defined point. Several usage scenarios will be investigated by the project, such as debugging, counting executions instancies and timing of program sections and memory monitoring.

- **Code smell detection**. Code smell detection helps in the refactoring of the existing software. Refactoring is the process of changing a software system in such a way that it does not alter the external behaviour of the code, while it improves its internal structure. It helps in introducing a disciplined way to ‘clean-up’ the code, while improving the code design during development and reducing the risks associated with a fast development of the code. The tool will reveal the most common design mistakes, called ‘smells’ so that they can be analysed and reduced.

3 Distributed computing and the Grid

3.1 Introduction

The ALICE computing model assumes the existence of a functional Grid middleware allowing for efficient, seamless, and democratic access to worldwide-distributed heterogeneous computing and storage resources. The technical feasibility of this assumption has been demonstrated during the series of Physics Data Challenges (PDCs) in 2003 and 2004 with the ALICE-developed AliEn [1] system.

Our current plan is to build a similar system making maximum usage of the common Grid services deployed by LCG [2] and adding the necessary ALICE-specific services derived from the AliEn system.

The experience gained during the data challenges strongly supports a Grid model with minimum of intrinsic hierarchy and specific resources categories. Intelligent workload scheduling based on the advertised features of the computing resources and on the job requirements allows for a better utilisation of the resources with respect to a fixed hierarchy. However, for clarity and to account for the possibility to have less advanced Grid services, the foreseen computational tasks and classes are categorized below in a manner that caters to a more commonly accepted 'tiered' distributed computing centre architecture.

The tier structure allows for a layered classification whereby, depending on the type of computational tasks, tasks are assigned to a computing element with a specific functionality. However, it also introduces a more rigid access to the resources, requiring a higher level of central management and thus increases the overall cost of the final system in terms of CPU, storage and manpower.

In this chapter a short description of the ALICE distributed computing environment is presented, with emphasis on the ALICE view of a decentralized Grid structure with advanced functionality. Owing to the rapid evolution of the Grid, it is expected that the distributed computing environment will undergo several modifications, especially in the area of the Grid implementation details.

3.2 Distributed computing

The ALICE computing model is driven by the consideration of the large amount of computing resources which will be necessary to store and process the data generated by the experiment. Detailed description of the data sizes under different data-taking scenarios are given in Chapter 7.

Since the conceptual design of the LHC [3] experimental programme, it was recognized that the required data processing and storage resources cannot be consolidated at a single computing centre. It is more natural, considering both the substantial financial investment involved and the need for expert human resources, that these resources are distributed at the HEP computing facilities of the institutes and universities participating in the experiment. The technical side of the decentralized offline computing scenario has been formalized in the so-called MONARC model [4] schematically shown in Fig. 3.1. MONARC describes an assembly of distributed computing resources, concentrated in a hierarchy of centres called Tiers, where Tier 0 is CERN, Tier 1s are the major computing centres which provide a safe data storage, likely in the form of a mass storage system (MSS), Tier 2s are smaller regional computing centres. The MONARC model also foresees Tier 3s which are university departmental computing centres and Tier 4s that are user workstations; however the distinction of these lower Tiers is not relevant in a Grid model. The major difference between the first three Tiers is the quality of service (QoS) and reliability of the computing resources at every level, where the highest QoS is offered by the Tier 0/Tier 1 centres. At the moment of publication of this document, the QoS metrics associated to the different Tiers are still under discussion.

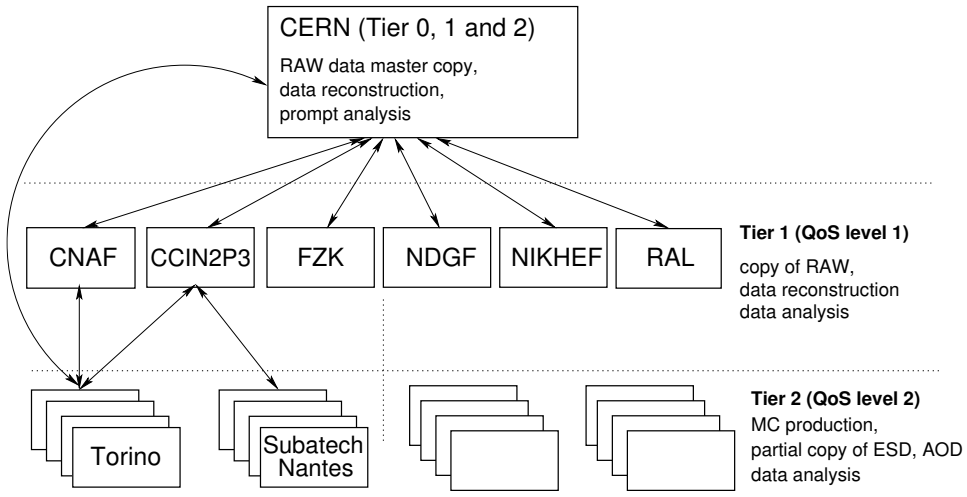

Figure 3.1: Schematic view of the ALICE offline computing tasks in the framework of the tiered MONARC model. The acronyms are explained in the text.

The basic principle underlying the ALICE computing model is that every physicist should have equal access to the data and the computing resources necessary for its processing and analysis. Thus, the resulting system will be very complex with hundreds of components at each site with several tens of sites. A large number of tasks will have to be performed in parallel, some of them following an ordered schedule, for example the raw data reconstruction, large Monte Carlo production, data filtering and stripping, and some being largely unpredictable: single-user Monte Carlo production and data analysis. In order to be used efficiently, the distributed computing and storage resources will have to be transparent to the end user, essentially looking like a single entity.

The commonality of distributed resources management is being realized under the currently ongoing development of the Grid [5]. It was conceived to facilitate the development of new applications based on high-speed coupling of people, computers, databases, instruments, and other computing resources by allowing "dependable, consistent, pervasive access to high-end resources". Although the MONARC model pre-dates the appearance of the Grid concept, its terminology is well adapted to the distribution of resources that are present in the HEP community and remains very useful for discussing the organization and relations of the centres. In this document, we shall consider only Tier 0, Tier 1 and Tier 2 functionality.

However, in a well functioning Grid, the distribution of tasks to the various computing centres will be performed dynamically, based on the availability of resources and the services that they advertise, irrespective of the assigned Tier level. This picture is an evolution of the MONARC structure, since it is more flexible and allows for a better optimisation of resource usage. This Tier-free model is sometimes called 'Cloud Model' and it has been adopted as a concept in the development of the AliEn system and used to exploit the computing capacities offered by the computing centres participating in the ALICE Grid during the Physics Data Challenges.

The ALICE computing model foresees that one copy of the raw data from the experiment will be stored at CERN (Tier 0) and a second copy will be distributed among the external (i.e. not at CERN) Tier 1 centres, thus providing a natural backup. Reconstruction to the Event Summary Data (ESD) level will be shared by the Tier 1 centres, with the CERN Tier 0 responsible for the first reconstruction pass. Subsequent data reduction to the Analysis Object Data (AOD) level, analysis and Monte Carlo production will be a collective operation where all Tier 1 and 2 centres will participate. The Tier 1s will perform reconstruction and scheduled analysis, while the Tier 2s will perform Monte Carlo and end-user analysis.

Grid technology holds the promise of greatly facilitating the exploitation of LHC data for physics

research and ALICE is very active on the different Grid test-beds and worldwide projects [2], [6], where Grid middleware prototypes are deployed. The objective of this activity is to gain experience with all systems, representing a reasonable amount of computing resources, and ultimately assemble a computing environment which is able to fulfil the collaboration needs in terms of offline data production and analysis.

This activity, both very useful and interesting in itself, is faced with the different levels of maturity and functionality of the deployed middleware and its various flavours emerging from different Grid projects and communities. Any middleware currently available is largely a result of leading-edge computer science research and is therefore still rather far from production quality. Moreover, there are no commonly adopted inter-Grid communication standards and the development in different projects is following very different paths and software standards, even if the functionality which is aimed at is very similar.

The emerging heterogeneous picture is rather unfavourable, since the computing resources to be exploited by the experiments like ALICE are presented in the form of non-conformal Grid implementations. In addition, owing to the multitude of projects working independently on a solution to essentially the same problems, the most complex tasks to be executed on the Grid, e.g. distributed data analysis, cannot be adequately addressed.

Faced with this situation, the LHC experiments are developing a very similar approach, although on parallel lines. They are planning to deploy services offering a homogeneus view of the Virtual Organisation corresponding to the experiment, while these services are in turn interfaced to the different Grid flavours deployed on the resources offered to the experiment by the funding agencies. In a non-dissimilar way, ALICE is planning to use some of the AliEn services to interface with the different Grids. It is of course in the best of ALICE's interest to minimise the amount of experiment specific services that we will have to deploy and maintain ourselves, maximising the use of the common Grid services deployed and maintained by the computing centres.

3.3 AliEn, the ALICE interface to the Grid

The AliEn (**Ali**CE **En**vironment) framework has been developed with the aim of offering to the ALICE user community a transparent access to computing resources distributed worldwide through a single interface. During the years 2001–2004 AliEn has provided a functional computing environment fulfilling the needs of the experiment in its preparation phase. AliEn was primarily conceived as the ALICE user entry point into the Grid world, shielding the users from its underlying complexity and heterogeneity. Through interfaces, it can use transparently resources of different Grids developed and deployed by other groups. This advanced concept has been successfully demonstrated during the ALICE Physics Data Challenge '04 (PDC'04), where the resources of the LCG and INFN Grids were accessed through interfaces. Approximately 11% of the computing resources used in PDC'04 were provided by the LCG and INFN Grids. In the future, the cross-Grid functionality will be extended to cover other Grid flavours. In addition, AliEn has been engineered to be highly modular, and individual components can be deployed and used in a foreign Grid, which is not adapted to the specific computational needs of ALICE, thus achieving an efficient use of the available resources.

The system is built around Open Source components and uses a Web Services model [7] and standard network protocols. Less than 5% is native AliEn code (mostly code in PERL), while the rest of the code has been imported in the form of Open Source packages and modules.

Web Services play the central role in enabling AliEn as a distributed computing environment. The user interacts with them by exchanging SOAP (Simple Object Access Protocol) messages and they constantly exchange messages between themselves behaving like a true Web of collaborating services. AliEn consists of the following components and services: authentication, authorization and auditing services; workload and data management systems; file and metadata catalogues; the information service; Grid and

job monitoring services; storage and computing elements. A schematic view of the AliEn services, their location and interaction with the native services at the computing centres is presented in Fig. 3.2.

The AliEn workload management system is based on the so-called 'pull' approach. A service manages a common task queue, which holds all the jobs of the ALICE Virtual Organization (VO). On each site providing resources for the ALICE VO, Computing Element (CE) services act as 'remote queues' giving access to computational resources that can range from a single machine, dedicated to running a specific task, to a cluster of computers in a computing centre, or even an entire foreign Grid. When jobs are submitted, they are sent to the central queue. The workload manager optimizes the queue taking into account job requirements such as the input files needed, the CPU time and the architecture requested, the disk space request and the user and group quotas. It then makes jobs eligible to run on one or more computing elements. The CEs of the active nodes get jobs from the central queue and deliver them to the remote queues to start their execution. The queue system monitors the job progress and has access to the standard output and standard error.

Input and output associated with any job are registered in the AliEn File Catalogue (FC), a virtual file system in which logical names, with a semantics similar to the Unix file system, are assigned to files. Unlike real file systems, the FC does not own the files; it only keeps an association between one or possibly more Logical File Names (LFN) and (possibly more than one) Physical File Names (PFN) on a real file or mass storage system. The correspondance is kept via the **G**lobal **U**nique file **ID**entifier (GUID) stored in the FC. The FC supports file replication and caching and it provides the information about file location to the RB when it comes to scheduling jobs for execution. These features are of particular importance, since similar types of data will be stored at many different locations and the necessary data replication is assumed to be provided transparently and automatically by the Grid middleware. The AliEn file system associates metadata with LFNs.

ALICE has used the system for distributed production of Monte Carlo data, reconstruction and analysis at over 30 sites on four continents. The round of simulation, reconstruction and analysis during the PDC'04 was aimed at providing large amounts of simulated data for physics studies as well as testing the main components of the ALICE computing model. During the data challenge, more than 400 000 jobs were successfully run worldwide from the AliEn Task Queue (TQ), producing 40 TB of data. Computing and storage resources were available both in Europe and the US. The amount of processing needed for

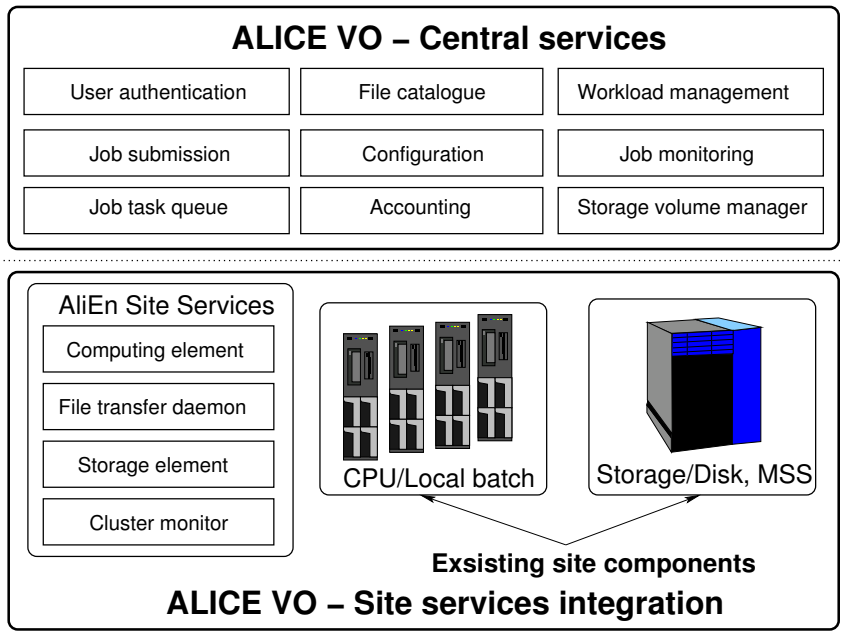

Figure 3.2: Schematic view of the AliEn basic components and deployment principles.

a typical production is in excess of 30 MSI2k×s to simulate and digitize a central Pb–Pb event. Some 100 000 high-multiplicity Pb–Pb events were generated for each major production. This is an average over a very large range since peripheral events may require one order of magnitude less CPU, and pp events two orders of magnitude less. The Pb–Pb events are then reprocessed several times superimposing known signals, in order to be reconstructed and analysed. Again there is a wide spread in the time this takes, depending on the event, but for a central event this needs a few MSI2k×s. Each Pb–Pb central event occupies about 2 GB of disk space, while pp events are two orders of magnitude smaller. The total amount of CPU work during PDC'04 was 750 MSI2k×h. The relative contribution of the computing centres, participating in the ALICE Grid during PDC'04 is shown in Fig. 3.3.

The Grid user data analysis has been tested in a limited scope using tools developed in the context of the ARDA project [8] (the gShell interface to the FC and the analysis tools based on it). Two approaches were prototyped and demonstrated: the asynchronous (interactive batch approach) and the synchronous (true interactive) analysis.

The asynchronous model has been realized by extending the ROOT [9] functionality to make it Grid-aware. As the first step, the analysis framework has to extract a subset of the datasets from the file catalogue using metadata conditions provided by the user. The next part is the splitting of the tasks according to the location of datasets. Once the distribution is decided, the analysis framework splits the job into sub-jobs and inserts them in the AliEn TQ with precise job descriptions. These are submitted to the local CEs for execution. Upon completion, the results from all sub-jobs are collected, merged and delivered to the user. This model has been shown to work with satisfactory results, but more attention has to be devoted to the optimisation of the load on the FC, when many simultaneous large user queries are performed and the fault-tolerance of the merging mechanism when some of the sub-jobs fail.

The synchronous analysis model requires a much tighter integration between ROOT and the Grid services, where the framework should be able to execute in parallel and in real-time all sub-jobs associated to the main user job. In addition, the system should automatically scale the number of running processes to the amount of available resources at the time of execution. The model relies on extending

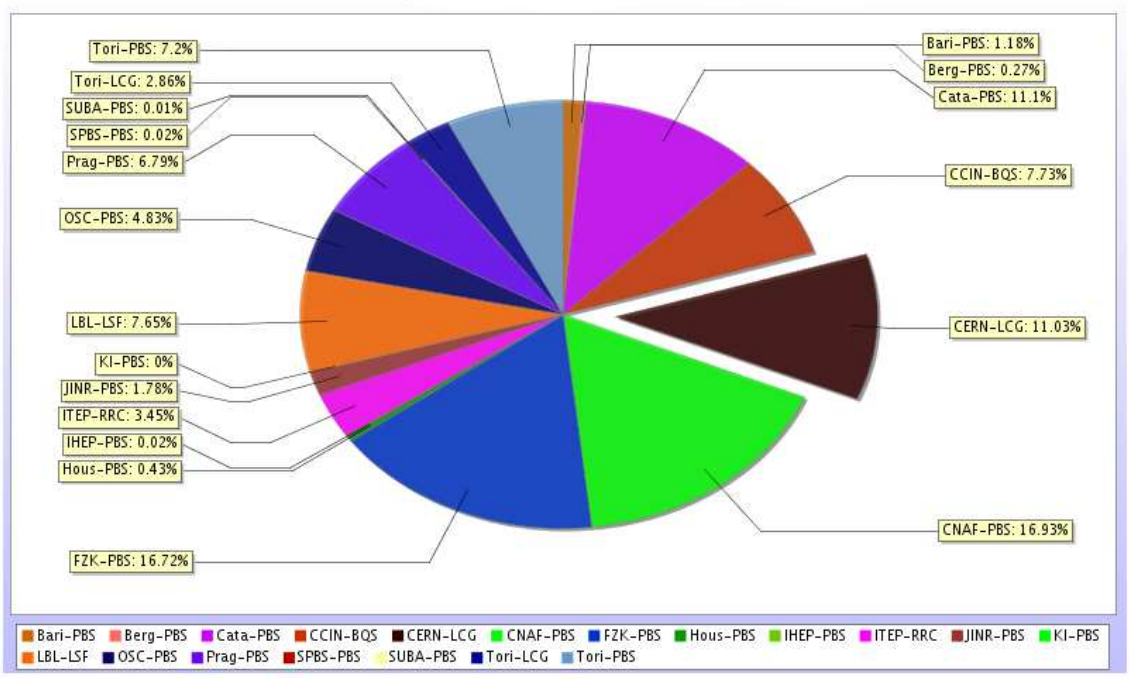

Figure 3.3: Relative CPU and storage contribution by the participating sites during the PDC'04.

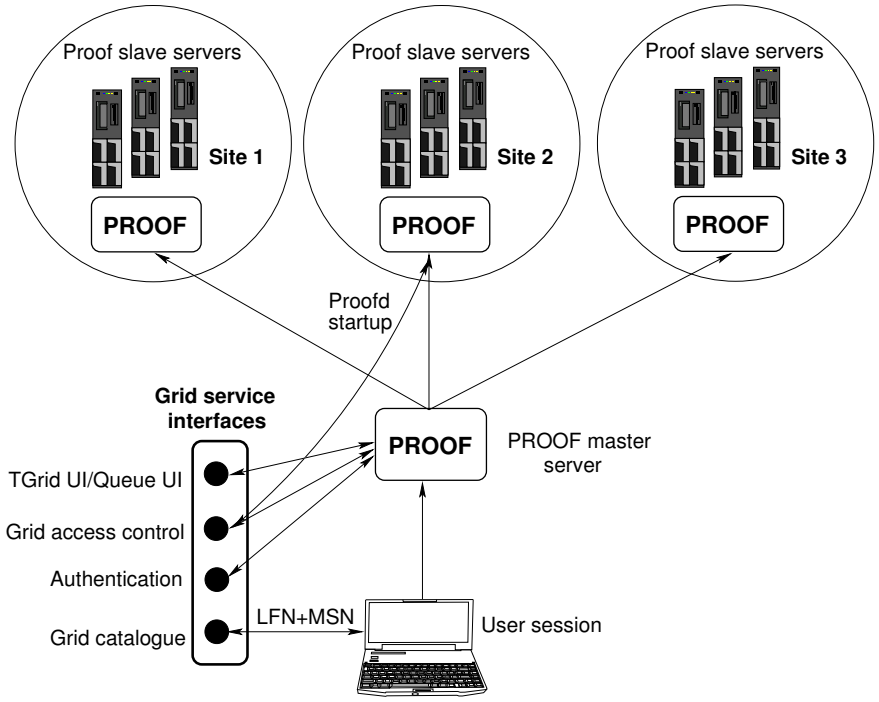

Figure 3.4: Setup and interaction with the Grid middleware of a user PROOF session distributed over many computing centres.

the functionality of PROOF [10] – the Parallel ROOT Facility. The PROOF interface to Grid-like services is presently being developed, focusing on authentication and the use of the FCs, in order to make both accessible from the ROOT shell.

In the conventional, single-site setup, PROOF workers are managed by a PROOF master server, which distributes tasks and collects results. In a multi-site setup, each site running a PROOF environment will be seen as a PROOF worker for a PROOF master running on the user machine. The PROOF master has therefore to implement the functionality of a master and a worker at the same time. This concept is illustrated in Fig. 3.4. AliEn classes used for asynchronous analysis as described earlier can be used for task splitting in order to provide the input data sets for each site that runs PROOF locally.

The AliEn-PROOF-based system for distributed synchronous analysis will be used for a rapid evaluation of large data samples in a time-constrained situation, for example the evaluation of the detector calibration and alignment at the beginning of a data-taking period. This will allow for an efficient planning of critical analysis tasks, where the predictability of the execution time is very important. As such it is an essential building block of the ALICE computing model.

3.4 Future of the Grid in ALICE

The experience with the AliEn Grid middleware has been instrumental in shaping the ALICE computing model. The technical feasibility of a functional Grid, effectively managing thousands of processors and hundreds of terabytes of storage, distributed over many computing centres worldwide has been demonstrated in a series of realistic Physics Data Challenges. These were conceived and executed with parameters closely approximating the real running conditions of the ALICE experiment.

One of the most important requirements for the efficient use of the Grid is the existence of a single interface to the computing resources, effectively shielding the end user from the underlying software and hardware complexity. In the past several years this role has been played by AliEn acting as a complete vertical Grid system.

During this time, different Grid solutions developed by large collaborations, have come to maturity. However, as explained before, two major issues still remain. First of all, none of these Grid flavours provides a complete solution for the ALICE computing model, and secondly there are no accepted standards, and all these Grids provide a different user interface and a diverse spectrum of functionality.

Therefore some of the AliEn services will continue to be used as the ALICE's single point of entry to the computing resources encapsulated by the various Grid entities and as a complement of their functionality to implement the ALICE computing model. In this model, which has already been prototyped and used, the foreign Grid will be accessed via interfaces. The elements of AliEn used in synergy with the deployed Grid middleware to supplement its functionality will assure efficient use of the computing resources. The cross-Grid role of AliEn preserves the present ALICE infrastructure and user access methods and allows for its continuous development and enrichment with advanced functionalities. It also provides the necessary methods for addition of computing resources and adoption of new Grid standards as they become available.

The ALICE Computing Project will closely watch the evolution of the functionality of the deployed Grid middleware and of the international Grid standards. Whenever a standard Grid service is found to provide the same, or better, functionality than an ALICE-specific one, its adoption will be considered in order to reduce the maintenance load and increase portability and robustness of the ALICE computing environment.

4 Simulation

4.1 Event generators

Heavy-ion collisions produce a very large number of particles in the final state. This is a challenge for the reconstruction and analysis algorithms which require a predictive and precise simulation of the detector response.

The ALICE experiment was designed when the highest nucleon–nucleon center-of-mass energy in heavy-ion interactions was at 20 GeV per nucleon–nucleon pair at the CERN SPS, i.e. a factor of about 300 less than the LHC energy. Model predictions, discussed in Volume 1 of the the ALICE Physics Performance Report [1], for the particle multiplicity in Pb–Pb collisions at LHC vary from 1400 to 8000 charged particles per rapidity unit at mid-rapidity. In summer 2000 the RHIC collider came online. The RHIC data seem to suggest that the LHC multiplicity will be on the lower side of the predictions. However, the RHIC top energy of 200 GeV per nucleon–nucleon pair is still 30 times less than the LHC energy. The extrapolation is so large that both the hardware and software of ALICE had to be designed to cope with the highest predicted multiplicity. On the other hand, we have to use different generators for the primary interaction, since their predictions are quite different at LHC energies.

The simulations of physical processes are confronted with several issues:

- Existing event generators give different predictions for the expected particle multiplicity, p_{t} and rapidity distributions, and the dependence of different observables on p_{t} and rapidity at LHC energies.
- Most of the physics signals, like hyperon production, high-p_{t} observables, open charm and beauty, quarkonia, etc. even at lower energies, are not exactly reproduced by the existing event generators.
- Simulation of small cross-section observables would demand prohibitively long runs to simulate a number of events that is commensurable with the expected number of detected events in the experiment.
- The existing generators do not simulate correctly some features like momentum correlations, flow, etc.

Nevertheless, to allow for efficient simulations we have developed the offline framework such that it allows for a number of options:

- The simulation framework provides an interface to several external generators, like for example HIJING [2] and DPMJET [3].
- A simple event generator based on parametrized η and p_{t} distributions can provide a signal-free event with multiplicity as a parameter.
- Rare signals can be generated using the interface to external generators like PYTHIA [4] or simple parametrizations of transverse momentum and rapidity spectra defined in function libraries.
- The framework provides a tool to assemble events from different signal generators (event cocktails).
- The framework provides tools to combine underlying events and signal events on the primary particle level (cocktail) and on the digit level (merging).
- Afterburners are used to introduce particle correlations in a controlled way.

The implementation of these strategies is described below. The theoretical uncertainty on the description of heavy-ion collisions at LHC has several consequences for our simulation strategy. A large part of the physics analysis will be the search for rare signals over an essentially uncorrelated background of emitted particles. To avoid being dependent on a specific model, and to gain in efficiency and flexibility, we generate events from a specially developed parametrization of a signal-free final state. This is based on a parametrization of the HIJING pseudo-rapidity (η) distribution and of the transverse momentum (p_t) distribution of CDF [5] data. To simulate the highest anticipated multiplicities we scale the η-distribution so that up to 8000 charged particles per event are produced in the range $|\eta| < 0.5$. Events generated from this parametrization are sufficient for a large number of studies, such as optimization of detector and algorithms performance, e.g. studies of track reconstruction efficiency as a function of particle multiplicity and occupancy. For physics performance studies, we have to simulate more realistic Pb–Pb collisions. HIJING, which yields charged particle multiplicities of up to $dN/d\eta \approx 6000$, is used to simulate the underlying event that is subsequently merged with different signals like jets or heavy flavour.

In order to facilitate the usage of different generators we have developed a generator base class called `AliGenerator`, see Fig. 4.1. Each `AliGenerator` implementation has to provide a basic set of configuration methods such as kinematics and vertex cuts. A high degree of flexibility is reached by providing several pointers as data members to:

- `TGenerator` in order to use external generators (see below);
- `AliVertexGenerator` for the possibility to obtain the event vertex from an external source needed for event merging;
- `AliCollisionGeometry` to provide a collision geometry (impact parameter, number of participants, etc.) to the event header or to other generators;
- `AliStack` to allow for the stand-alone use of a generator.

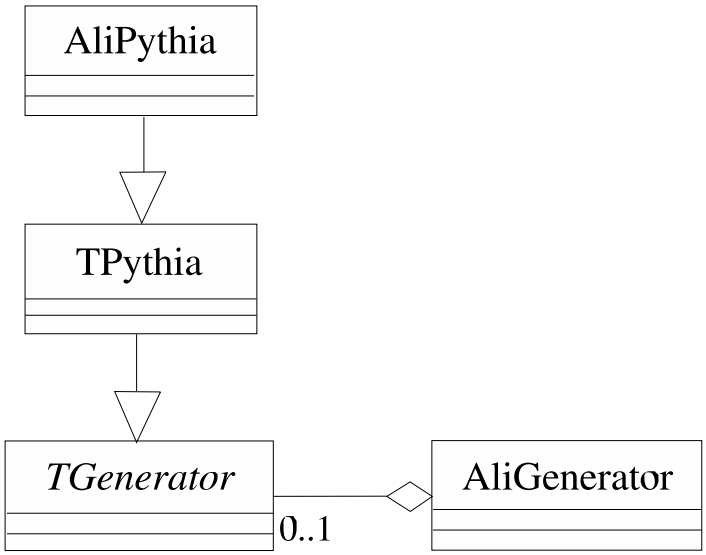

Figure 4.1: `AliGenerator` is the base class that has the responsibility of generating the primary particles of an event. Some realizations of this class do not generate the particles themselves but delegate the task to an external generator like PYTHIA through the `TGenerator` interface.

Several event generators are available via the abstract ROOT class that implements the generic generator interface, `TGenerator`. Through implementations of this abstract base class we wrap FORTRAN Monte Carlo codes like PYTHIA, HIJING, etc. that are thus accessible from the AliRoot classes. In particular, the interface to PYTHIA, `AliPythia` deriving from `TPythia`, includes the use of nuclear structure functions of PDFLIB and a large set of preconfigured processes such as jet-production, heavy flavours, and pp minimum bias.

In many cases, the expected transverse momentum and rapidity distributions of particles are known. In other cases the effect of variations in these distributions must be investigated. In both situations it is appropriate to use generators that produce primary particles and their decays sampling from parametrized spectra. To meet the different physics requirements in a modular way, the parametrizations are stored in independent function libraries wrapped into classes that can be plugged into the generator. This is schematically illustrated in Fig. 4.2 where four different generator libraries can be loaded via the abstract generator interface.

It is customary in heavy-ion event generation to superimpose different signals on an event to tune the reconstruction algorithms. This is possible in AliRoot via the so-called cocktail generator (Fig. 4.3). This creates events from user-defined particle cocktails by choosing as ingredients a list of particle generators.

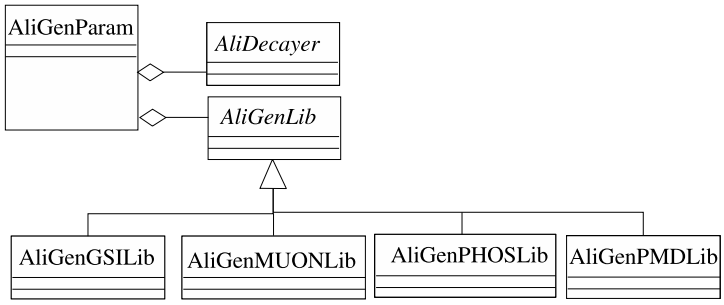

Figure 4.2: `AliGenParam` is a realization of `AliGenerator` that generates particles using parametrized p_t and pseudo rapidity distributions. Instead of coding a fixed number of parametrizations directly into the class implementations, user-defined parametrization libraries (AliGenLib) can be connected at run time allowing for maximum flexibility.

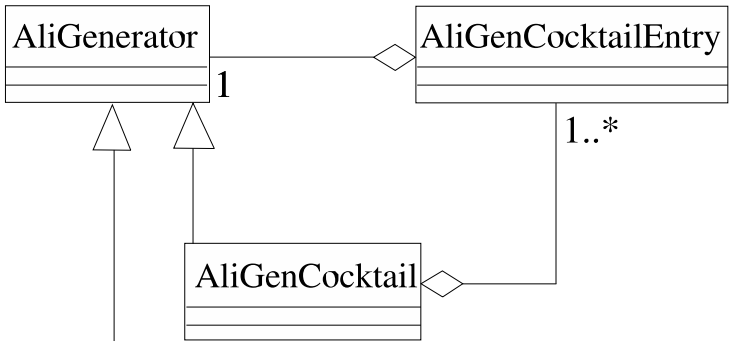

Figure 4.3: The `AliCocktail` generator is a realization of `AliGenerator` which does not generate particles itself but delegates this task to a list of objects of type `AliGenerator` that can be connected as entries (`AliGenCocktailEntry`) at run time. In this way different physics channels can be combined in one event.

4.2 Afterburner processors and correlation analysis

The modularity of the event generator framework allows easy integration with the simulation steering class `AliRun` and with the objects that are responsible for changing the output of event generators or for assembling new events making use of the input of several events. These processors are generally called 'afterburners'. They are especially needed to introduce a controlled (parametrized) particle correlation into an otherwise uncorrelated particle sample. In AliRoot this task is further simplified by the implementation of a stack class (`AliStack`) that can be connected to both `AliRun` and `AliGenerator`. Currently, afterburners are used for the simulation of the two-particle correlations, flow signals, and jet quenching.

4.3 Detector response simulation

To respond to the ALICE simulation requirements, it is important to have a high-quality and reliable detector response simulation code. One of the most common programs for full detector simulation is GEANT 3 [6] which, however, is a 20-year old FORTRAN program, not being (officially) further developed since 1993. GEANT 4 [7] is being developed by a large international collaboration with a strong component in CERN/IT as the OO simulation package for the LHC. We are also using FLUKA [8] as a full detector simulation program. These three programs have a very different user interface, therefore we decided to build an environment that could take advantage of the maturity and solidity of GEANT 3 and, at the same time, protect the investment in the user code when moving to a new Monte Carlo. In order to combine immediate needs and long term requirements into a single framework, we wrapped the GEANT 3 code in a C++ class (`TGeant3`) and we developed a Virtual Monte Carlo (VMC) abstract interface (now part of ROOT, see below). We have interfaced GEANT 4 and FLUKA with our virtual Monte Carlo interface. We will thus be able to change the simulation engine without any modification in the user detector description and signal generation code. This strategy has proved very satisfactory and we are able to assure coherence of the whole simulation process which includes the following steps regardless of the particle transport package in use:

- **Event generation of final-state particles:** The collision is simulated by a physics generator code or a parametrization and the final-state particles are fed to the transport program.
- **Particle transport:** The particles emerging from the interaction of the beam particles are transported in the material of the detector, simulating their interaction with it and the energy deposition that generates the detector response (hits). An event display is shown in Colour Figure II.
- **Signal generation and detector response:** During this phase the detector response is generated from the energy deposition of the particles traversing it. This is the ideal detector response, before the conversion to digital signals and the formatting of the front-end electronics is applied.
- **Digitization:** The detector response is digitized and formatted according to the output of the front-end electronics and the data acquisition system. The results resemble closely the real data that will be produced by the detector.
- **Fast simulation:** The detector response is simulated via appropriate parametrizations or other techniques that do not require the full particle transport.

Virtual Monte Carlo interface

As explained above, our strategy to isolate the user code from changes of the detector simulation package was to develop a virtual interface to the detector transport code. We call this interface Virtual Monte Carlo. It is implemented [9] via C++ virtual classes and is schematically shown in Figs. 4.4

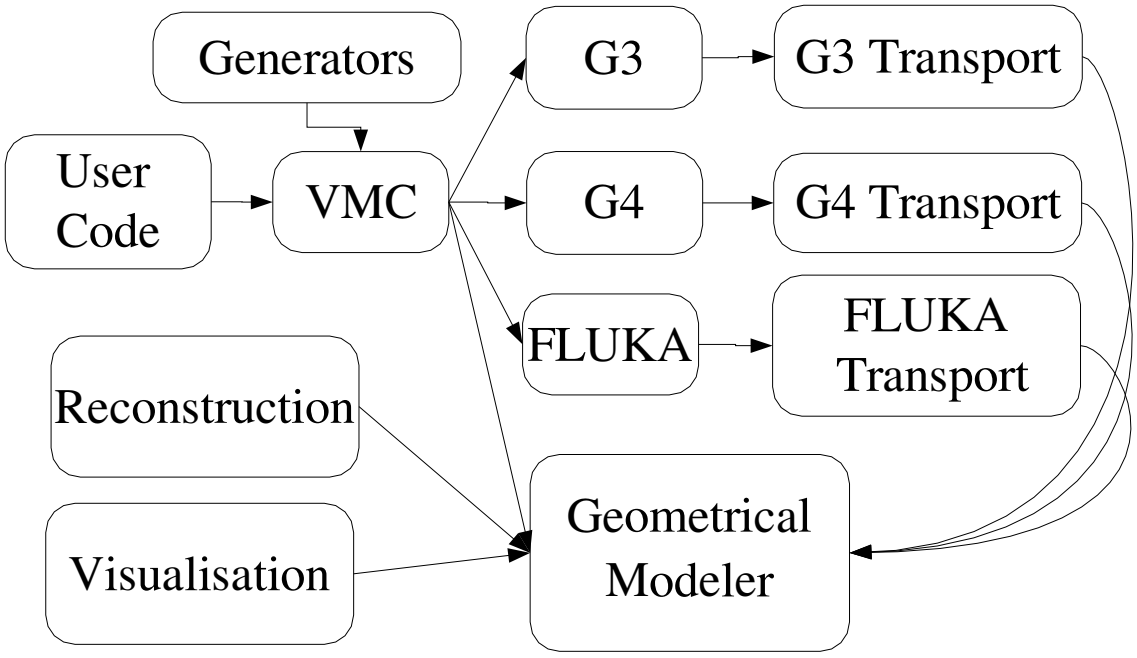

Figure 4.4: The Virtual Monte Carlo concept.

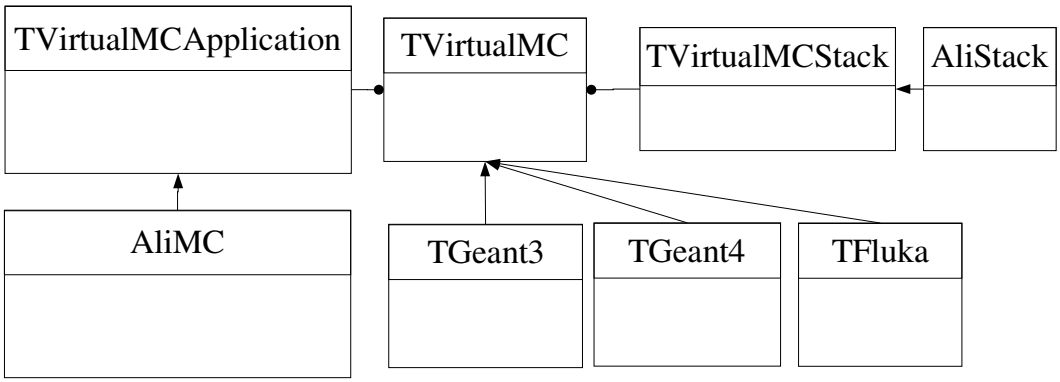

Figure 4.5: The Virtual Monte Carlo design and its realization within the AliRoot framework.

and. 4.5. The codes that implement the abstract classes are real C++ programs or wrapper classes that interface to FORTRAN programs.

An additional step is to replace the geometrical modeller of the different packages with a single one, independent from any specific simulation engine; the aim is to use the same geometrical modeller also for reconstruction and analysis. Thanks to the collaboration between the ALICE Computing project and the ROOT team, we have developed a geometrical modeller, `TGeo` [10], that is able to represent the ALICE detector, and to replace the GEANT 3 modeller for navigation in the detector. It has also been interfaced to FLUKA and discussions are under way with the GEANT 4 team to interface it to the GEANT 4 Monte Carlo. Using the `roottog4` converter, geometries defined for `TGeo` can be converted into GEANT 4 geometries.

Using the virtual Monte Carlo we have converted all FORTRAN user code developed for GEANT 3 into C++, including the geometry definition and the user scoring routines, `StepManager`. These have been integrated in the detector classes of the AliRoot framework. The output of the simulation is saved directly with ROOT I/O, simplifying the development of the digitization and reconstruction code in C++.

GEANT

GEANT 3 is the detector simulation Monte Carlo code used extensively so far by the HEP community for simulation of the detector response. However, it is no longer maintained and has several known drawbacks, both in the description of physics processes, particularly hadronic [11], and of the geometry. Its designated successor is GEANT 4. ALICE has spent considerable effort in evaluating GEANT 4 via several benchmarks; details on ALICE experience with GEANT 4 can be found in Ref. [12]. We were able to keep the same geometry definition using the `G3toG4` utility to translate from GEANT 3 to GEANT 4 geometry; in addition we have improved `G3toG4` and made it fully operational. The virtual Monte Carlo interface allows us to run full ALICE simulations also with GEANT 4 and to compare them with the GEANT 3 results; the advantage being that both simulation programs use the same geometry and the same scoring routines. An additional advantage is a substantial economy of effort. Using the same geometry description eliminates one of the major sources of uncertainty and errors in the comparison between different Monte Carlos, which comes from the fact that it is rather difficult to make sure that, when comparing two Monte Carlos on a particular experimental configuration, there are no differences in the geometry description and in the scoring.

This exercise has exposed the GEANT 4 code to a real production environment and we experienced several of its weaknesses. We faced several problems with its functionality that have required substantial user development. In particular, the definition of volume or material-specific energy thresholds and mechanism lists are not so straightforward as in GEANT 3. The strategy of GEANT 3 was to provide the user one state-of-the-art implementation of the physics processes together with a few configuration options essentially for performance optimization and debugging. In contrast, GEANT 4 has to be configured using so-called physics lists corresponding to different combinations of model implementation that in addition may vary from detector to detector. They need a great amount of inside knowledge to be used correctly.

We have also performed a number of benchmark tests of the hadronic [13] and low-energy neutron transport [14].

We are now planning to interface GEANT 4 with the ROOT geometrical modeller to avoid the conversion step via `G3toG4` and to take advantage from its advanced solid modelling capabilities.

FLUKA

FLUKA plays a very important role in ALICE for all the tasks where detailed and reliable physics simulation is vital, given its thorough physics validation and its almost unique capability to couple low-energy neutron transport with particle transport in a single program. These include background calculation, neutron fluence, dose rates, and beam-loss scenarios [15]. An example for a neutron fluence map obtained with FLUKA is shown in Colour Figure III. FLUKA has been particularly important for ALICE in the design of the front absorber and beam shield. To ease the input of the FLUKA geometry, ALICE has developed an interactive interface [16], called `ALIFE`, that allows setups described with FLUKA to be combined and modified easily. Figure 4.6 schematically describes the use of `ALIFE` to prepare the input for FLUKA.

To provide another alternative to GEANT 3 for full detector simulation, we have developed an interface with FLUKA, again via the Virtual Monte Carlo. The native FLUKA geometry modeller has been replaced by `TGeo` which allows us to run FLUKA with the same geometry as used for GEANT 3 and GEANT 4 simulations.

Rigorous tests have been performed to ensure that the FLUKA native geometry modeller and navigator and the FLUKA interfaces to `TGeo` give exactly the same results. Based on these tests, the FLUKA Project considers the FLUKA-`TGeo` as validated.

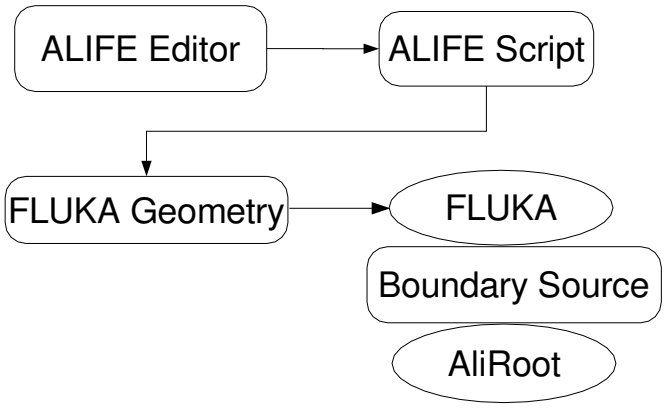

Figure 4.6: Example of the use of ALIFE. The ALIFE editor allows easy creation of an ALIFE script, which is in fact FLUKA input geometry. FLUKA is then used to transport particles, including low-energy neutrons, to a virtual boundary surface. The particles are then written to a file that is used as a source for a regular AliRoot simulation to evaluate the detector response to background.

4.3.1 Simulation framework

The AliRoot simulation framework can provide data at different stages of the simulation process [17], as described in Fig. 2.2 on page 16. Most of the terminology comes from GEANT 3. First, there are the so-called hits that represent the precise signal left by the particle in the detector before any kind of instrumental effect, i.e. precise energy deposition and position. These are then transformed into the signal produced by the detector, summable digits that correspond to the raw data before digitization and threshold subtraction. The introduction of summable digits is necessary because of the embedding simulation strategy elaborated for the studies in the Physics Performance Report. These summable digits are then transformed into digits that contain the same information as raw data, but in ROOT structures. The output of raw data in DATE (the ALICE data acquisition system [18]) format has already been done during the data challenges.

The ALICE detector is described in great detail, see Fig. 4.7, including services and support structures, beam pipe, flanges, and pumps. The AliRoot geometry follows the evolution of the baseline design of the detector in order to continuously provide the most reliable simulation of the detector response. AliRoot is also an active part of this process since it has been used to optimize the design, providing different geometry options for each detector. The studies that provided the results presented in the PPR were performed with the baseline geometry.

4.3.2 Geometry of structural elements

The description of the front- and small-angle absorber regions is very detailed on account of their importance to the muon spectrometer. The simulation has been instrumental in optimising their design and in saving costs without a negative impact on the physics performance. The material distribution and magnetic fields of the L3 solenoidal magnet and of the dipole magnets are also described in detail. The magnetic field description also includes the interference between the two fields. The field distributions are described by three independent maps for 0.2, 0.4 and 0.5 T solenoid L3 magnetic field strengths. Alternatively, it is possible to use simple parametrizations of the fields, i.e. constant solenoidal field in the barrel and a dipole field varying along the z direction for the muon spectrometer. The space frame, supporting the barrel detectors, is described according to its final design taking into account modifications to the initial design such that it allows the eventual addition of a proposed electromagnetic calorimeter [19].

The design of the ALICE beam pipe has also been finalized. All elements that could limit the detector performance (pumps, bellows, flanges) are represented in the simulation.

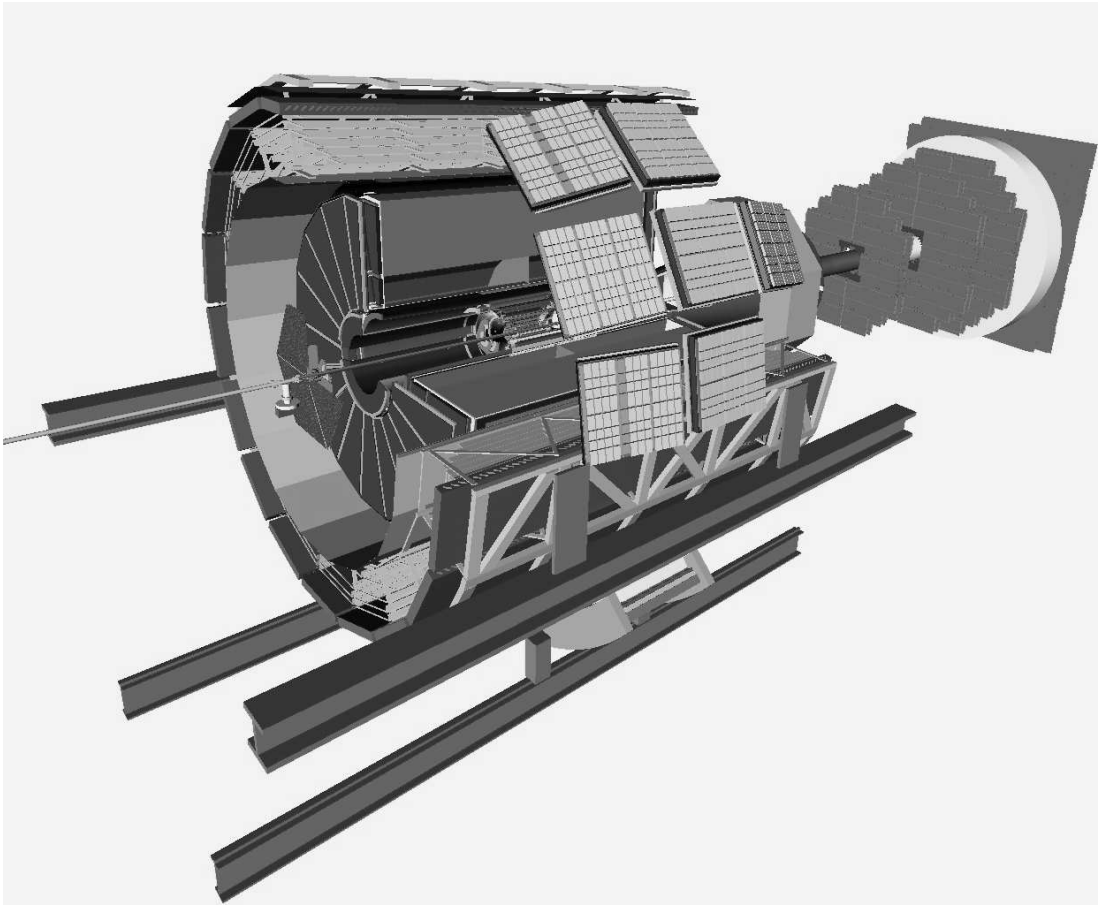

Figure 4.7: AliRoot simulation of the ALICE detector.

4.3.3 Geometry of detectors

Most of the detectors are described by two versions of their geometry; a detailed one, which is used to accurately simulate the detector response and study their performance, and a coarse version that provides the correct material budget with minimal details, and is used to study the effect of this material budget on other detectors. For some detectors, different versions of the geometry description corresponding to different geometry options are selectable via the input C++ script at run time. In the following we give some examples. We remind the reader that some of this information is still subject to rapid evolution.

Both a detailed and a coarse geometry are available for the ITS. The detailed geometry of the ITS is very complicated (see Colour Figure IV) and crucially affects the evaluation of impact parameter and electron bremsstrahlung. On the other hand, simulation of the coarse geometry is much faster when ITS hits are not needed.

Three configurations are available for the TPC. Version 0 is the coarse geometry, without any sensitive element specified. It is used for the material budget studies and is the version of interest for the outer detectors. Version 1 is the geometry version for the Fast Simulator. The sensitive volumes are thin gaseous strips placed in the Small (S) and Large (L) sectors at the pad-row centres. The hits are produced whenever a track crosses the sensitive volume (pad-row). The energy loss is not taken into account. Version 2 is the geometry version for the slow simulator. The sensitive volumes are S and L sectors. One can specify as sensitive volumes either all sectors or only a few of them, up to 6 S and 12 L sectors. The hits are produced in every ionizing collision. The transport step is calculated for every collision from an exponential distribution. The energy loss is calculated from an $1/E^2$ distribution and the response is parametrized by a Mathieson distribution.

The TRD geometry is simulated in great detail, including the correct material budget for electronics and cooling pipes. The full response and digitization have been implemented allowing studies of open questions such as the number of time-bins, the 9- or 10-bit ADC, the gas and electronics gain, the drift velocity, and maximum Lorentz angle. The transition-radiation photon yield is approximated by an analytical solution for a foil stack, with adjustment of the yield for a real radiator, including foam and fibre layers from test beam data. This is quite a challenging detector to simulate, as both normal energy loss in the gas and absorption of transition-radiation photons have to be taken into account. During the signal generation several effects are taken into account: diffusion, 1-dimensional pad response, gas gain and gain fluctuations, electronics gain and noise, as well as conversion to ADC values. Absorption and $\mathbf{E} \times \mathbf{B}$ effects will be introduced.

A detailed study of the background coming from slow neutron capture in Xe gas was performed [20] with FLUKA. The spectra of photons emitted after neutron capture are not included in standard neutron-reaction databases. An extensive literature search was necessary in order to simulate them. The resulting code is now part of the FLUKA Monte Carlo [21].

The TOF detector covers a cylindrical surface of polar acceptance $|\theta - 90^\circ| < 45^\circ$. It has a modular structure corresponding to 18 sectors in φ and to 5 segments in z. All modules have the same width of 128 cm and increasing lengths, adding up to an overall TOF barrel length of 750 cm. Inside each module the strips are tilted, thus minimizing the number of multiple partial-cell hits due to the obliqueness of the incidence angle. The double stack-strip arrangement, the cooling tubes, and the material for electronics have been described in detail. During the development of the TOF design several different geometry options were studied, all highly detailed.

The HMPID detector also poses a challenge in the simulation of the Cherenkov effect and the secondary emission of feedback photons. A detailed simulation has been introduced for all these effects and has been validated both by test-beam data and with the ALICE RICH prototype that has been operating in the STAR experiment. An event display with the typical rings is shown in Colour Figure V.

The PHOS has also been simulated in detail. The geometry includes the Charged Particle Veto (CPV), crystals (EMC), readout (APD) and support structures. Hits record the energy deposition in one CPV and one EMC cell per entering particle. In the digits the contribution from all particles per event are summed up and noise is added.

The simulation of the ZDC in AliRoot requires transport of spectator nucleons with Fermi spread, beam divergence, and crossing angle for over 100 m. The HIJING generator is used for these studies taking into account the correlations with transverse energy and multiplicity. Colour Figure VI shows the result of the simulation of the hadronic shower induced by a 2.7 TeV neutron.

The muon spectrometer is composed of five tracking stations and two trigger stations for which detailed geometries have been implemented. Supporting frames and support structures are coarsely described but they are not very important in the simulation of the signal. The muon chambers have a complicated segmentation that has been implemented during the signal generation via a set of virtual classes. This allows one to change the segmentation without modifying the geometry.

Summable digits (pad hits) are generated taking into account the Mathieson formalism for charge distribution, while work is ongoing on the angular dependence, Lorentz angles and charge correlation.

The complex T0–FMD–V0–PMD forward detector system is still under optimization. Several options are provided to study their performance.

The description of ALICE geometry and the generation of simulated data are in place. Hence the offline framework allows the full event reconstruction including the main tracking devices. The framework also allows comparison with test-beam data. The early availability of a complete simulation has been an important point for the development of reconstruction and analysis code and user interfaces, which is now the focus of the development.

4.4 Fast simulation

Owing to the expected high particle multiplicity for heavy-ion collisions at the LHC, typical detector performance studies can be performed with a few thousand events. However, many types of physics analysis, in particular of low cross-section observables, such as D meson reconstruction from hadronic decay channels, have to make use of millions of events. Computing resources are in general not available for such high-statistics simulations.

To reach the required sample size, fast simulation methods based on meaningful parametrizations of the results from detailed and consequently slow simulations are applied. The systematic error introduced by the parametrizations is in general small compared with the reduction of the statistical error. This is particularly true for the studies of the invariant-mass continuum below a resonance (cocktail plots).

It is hard to find a common denominator for fast simulation methods since they are very specific to the analysis task. As a minimum abstraction, we have designed base classes that allow for a representation of the detector or detector systems as a set of parametrizations of acceptance, efficiency, and resolution. The Muon Spectrometer fast simulation has been implemented using these classes.

Another interesting development concerns the fast simulation of the resolution and efficiency of track reconstruction in the central barrel. In this approach, resolution and efficiency in TPC are obtained from the track parameters at the inner radius of the TPC, using a parametrization. After this, full track reconstruction is performed for the inner tracking system, which is needed for detailed secondary vertex reconstruction studies. For details see Ref. [22].

4.5 Event merging and embedding

The simulation of small cross-section observables would demand prohibitively long runs to simulate a number of events commensurable with the expected number of detected events in the experiment. To circumvent this problem we use an event merging procedure: during digitization we produce so called summable digits. These are digits before the addition of electronic noise and pedestal subtraction. Merged events are produced by adding the summable digits from background and signal events. Each background event is used n times for merging. Since computing time is dominated by the simulation of the background events, in this way the event statistics is increased by a factor of n.

A similar technique called embedding consists in mixing data from simulation with real events. This allows for the realistic evaluation of track reconstruction performance in a high-particle-density environment. Here, events are mixed on the level of ADCs and are subsequently passed to the standard reconstruction code.

5 Reconstruction

5.1 Organization of the reconstruction code

The ALICE reconstruction code is part of the AliRoot framework. Its modular design allows its code to be compiled into separate shared libraries and executed independently on the other parts of AliRoot. As an input, the reconstruction uses the digits, i.e. ADC or TDC counts together with some additional information like module number, readout channel number, time bucket number, etc. The reconstruction can use both digits in a special ROOT format, more convenient for development and debugging purposes, and digits in the form of raw data, as they are output from the real detector or can be generated from the simulated special-format digits above (see Fig. 5.1). The output of the reconstruction is the Event Summary Data (ESD) containing the reconstructed charged particle tracks (together with the particle identification information), decays with the V^0 (like $\Lambda \rightarrow p\pi$), kink (like charged $K \rightarrow \mu\nu$) and cascade (like $\Xi \rightarrow \Lambda\pi \rightarrow p\pi\pi$) topologies and some neutral particles reconstructed in the calorimeters.

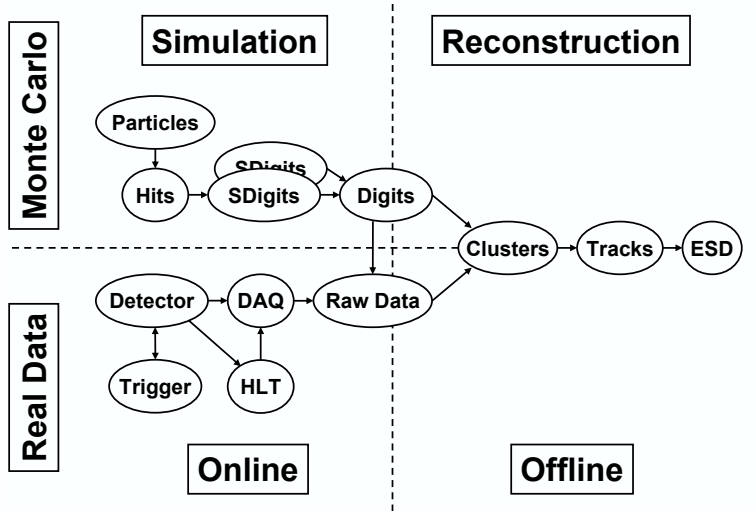

Figure 5.1: Interaction of the reconstruction code with the other parts of AliRoot.

The main steering reconstruction class, `AliReconstruction`, provides a simple user interface to the reconstruction. It allows users to configure the reconstruction procedure, include or exclude from the run a detector, and ensure the correct sequence of the reconstruction steps:

- reconstruction steps that are executed for each detector separately (typical example is the cluster finding);
- primary vertex reconstruction;
- track reconstruction and particle identification (PID);
- secondary vertex reconstruction (V^0, cascade and kink decay topologies).

The `AliReconstruction` class is also responsible for the interaction with the AliRoot I/O sub-system and the main loop over the events to be reconstructed belongs to this class too.

The interface from the steering class `AliReconstruction` to the detector-specific reconstruction code is defined by the base class `AliReconstructor`. For each detector there is a derived reconstructor

class. The user can set options for each reconstructor in the form of a string parameter. Detector-specific reconstructor classes are responsible for creating the corresponding specific cluster-, track- and vertex-finder objects and for passing the corresponding pointers to the `AliReconstruction`. This allows one to configure the actual reconstruction process using different versions of the reconstruction classes at the detector level.

The detailed description of the reconstruction in all the ALICE detectors can be found in Ref. [1]. Here we shall only outline briefly the most challenging parts of it.

5.2 Track reconstruction in the central detectors

A charged particle going through the detectors leaves a number of discrete signals that measure the position of the points in space where it has passed. These space points are reconstructed by a detector-specific cluster-finding procedure. For each space point we also calculate the uncertainty of the space-point position estimation. All of the central tracker detectors (ITS, TPC, TRD) have their own detailed parametrization of the space-point position uncertainties, however, some of the parameters can be fixed only at the track finding step (see below). The space points together with the position uncertainties are then passed to the track reconstruction. If, in addition to the space point position, the detector is also able to measure the produced ionization, this information can be used for the particle identification.

Offline track reconstruction in ALICE is based on the Kalman filter approach [2]. The detector specific implementations of the track reconstruction algorithm use a set of common base classes, which makes it easy to pass tracks from one detector to another and test various parts of the reconstruction chain. For example, we can easily switch between different implementations of the clustering algorithm or the track seeding procedure. This also allows us to use smeared positions of the simulated hits instead of the ones reconstructed from the simulated detector response, which is very useful for testing purposes. In addition, each hit structure contains the information about the track that originated it. Although, this implies the storage of extra information, it was proved to be very useful for debugging the track reconstruction code.

The event reconstruction starts with the determination of the position of the primary vertex. This can be done prior to track finding by a simple correlation of the space points reconstructed at the two pixel layers of the ITS. As was demonstrated in Ref. [3], the precision of ~ 5 μm along the beam direction and about 25 μm in the transverse plane is routinely achieved for the high multiplicity events. The information about the primary vertex position and its position uncertainty is then used during the track finding (seeding and applying the vertex constraint) and for the secondary vertex reconstruction.

The combined track finding in the central ALICE detectors consists of three passes that are described below (see also Fig. 5.2).

Initial inward reconstruction pass. The overall track finding starts with the track seeding in the outermost pad rows of the TPC. Different combinations of the pad rows are used with and without a primary vertex constraint. Typically more than one pass is done, starting with a rough vertex constraint, imposing the primary vertex with a resolution of a few centimetres and then releasing the constraint.

At first, we implemented the TPC track finding in a classical approach where cluster finding precedes the track finding. In addition, we also developed another approach where we defer the cluster finding at each pad row until all track candidates are propagated into its position. This way we know which of the clusters are susceptible to be overlapped and we may attempt cluster deconvolution at that specific place. In both approaches the track candidates are propagated and new clusters assigned to them using Kalman filtering.

Then, for each track reconstructed in the TPC, we search for its prolongation in the ITS. In the case of high-multiplicity events this is done by investigating a whole tree of possible track prolongations. First, we impose a rather strict vertex constraint with a resolution of the order of 100 μm or better. If a

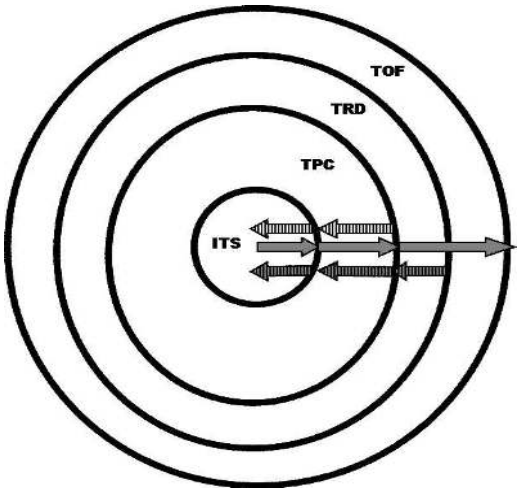

Figure 5.2: Schematic view of the three passes of the combined track finding (see the text).

prolongation is found, the track is refitted releasing the constraint. If the prolongation is not found we try another pass, without the vertex constraint, in order to reconstruct the tracks coming from the secondary vertices well separated from the main interaction point.

We thus obtain the estimates of the track parameters and their covariance matrix in the vicinity of the interaction point. At this moment we can also tell which tracks are likely to be primary. This information is used in the subsequent reconstruction steps.

Outward reconstruction pass and matching with the outer detectors. From the innermost ITS layer we proceed with the Kalman filter in the outward direction. During this second propagation we remove from the track fit the space points with large χ^2 contributions. In this way we obtain the track parameters and their covariance matrix at the outer TPC radius. We continue the Kalman filter into the TRD and then match the tracks toward the outer detectors: the TOF, HMPID, PHOS and EMCAL.

When propagating the primary track candidates outward, we also calculate their track length and time of flight for several mass hypotheses in parallel. This information is needed for the PID in the TOF detector.

Final reconstruction pass. After the matching with the outer detectors, all the available PID information is assigned to the tracks. However now the track momenta are estimated far away from the primary vertex. The task of the final track reconstruction pass is to refit the primary tracks back to the primary vertex or, in the case of the secondary tracks, as close to the vertex as possible. This is done again with the Kalman filter using in all the detectors the clusters already associated at the previous reconstruction passes. During this pass we also reconstruct the secondary vertices (V^0s, cascade decays and kinks).

The whole procedure is completed with the generation of the ESD. A typical ESD for a central Pb–Pb event contains about 10^4 reconstructed tracks, a few hundred V^0 and kink candidates, and a few tens of cascade particle candidates.

Performance of the track reconstruction. Figure 5.3 shows the efficiency of the combined track reconstruction as a function of particle momentum for events with different multiplicities. The track reconstruction efficiency is defined as a ratio of the number of reconstructed 'good' tracks to the number of 'good' generated tracks (for the definition of the 'good', in other words 'reconstructable', tracks see [4]).

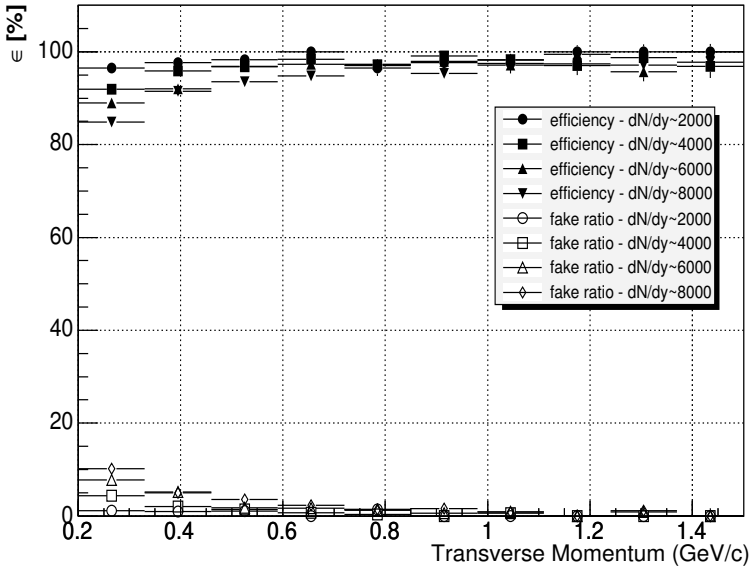

Figure 5.3: Combined track reconstruction efficiency (closed symbols) and probability of obtaining a fake track (open symbols) as a function of transverse momentum for different track multiplicities.

Some of the reconstructed tracks can be associated with a certain number of clusters which do not belong to those tracks. The probability of obtaining such tracks ('fake' tracks) is also shown in this picture. One can see that even in the case of events with the highest expected multiplicity, the track-finding efficiency is always above 85% and the number of the 'fake' tracks never exceeds a few per cent.

The momentum resolution obtained with different detector configurations and different versions of the reconstruction in the TRD is shown in Fig. 5.4. In all the cases the resolution at 100 GeV/c is better than 5%.

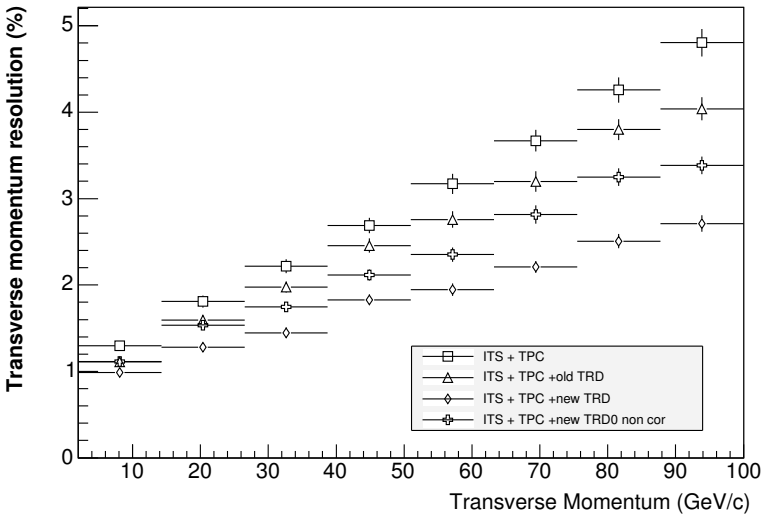

Figure 5.4: Momentum resolution as a function of particle momentum for high-momentum tracks and different detector configurations.

5.3 Track reconstruction in the forward muon spectrometer

Since the muon spectrometer geometry is quite different from that of the central detectors (notably, the large distance (up to 2.5 m) between consecutive measurements), it was not obvious from the beginning that the Kalman filter would demonstrate the best performance possible as compared with other methods. That is why another algorithm was developed originally which further served as a reference point for Kalman filter studies.

This original method was motivated by the Kalman filter strategy, i.e. implements a simultaneous track finding and fitting approach as follows. Track candidates start from segments (vectors) found in the last two tracking stations, where a segment is built from a pair of points from two chamber planes of the same tracking station. Then each track is extrapolated to the first station, and segments or single hits found in the other stations are added sequentially. For each added station, the track candidate is refitted and the hits, giving the best fit quality, are kept. In order to increase the track-finding efficiency the procedure looks for track continuation in the first two stations in direct and reverse order. A track is validated if the algorithm finds at least 3 hits (out of 4 possible) in the detector planes behind the dipole magnet, at least 1 hit (out of 2) in the station located inside the magnet and 3 hits (out of 4) in the chambers before the magnet.

For a Kalman track reconstruction, tracks are initiated for all track segments found in the last two detector stations as for the previous method. The tracks are parametrized as $(y, x, \alpha, \beta, q/p)$, where y is a coordinate in the bending plane, x is a non-bending coordinate, α is a track angle in the bending plane with respect to the beam line, β is an angle between the track and the bending plane, q and p are the track charge and momentum, respectively.

A track starting from a seed is followed to the first station or until it is lost (if no hits in a station are found for this track) according to the following procedure. It propagates the track from the current z-position to a hit with the nearest z-coordinate. Then for given z it looks for the hits within a certain window w around the transverse track position. After this there are two possibilities. The first one is to calculate the χ^2-contribution of each hit and consider the hit with the lowest contribution as belonging to the track. The second way is to use the so-called track branching and pick up all the hits inside the acceptance window. Efficiency and mass resolution tests have shown that the second way gives a better result.

After propagation to the chamber 1 all tracks are sorted according to their quality Q, defined as

$$Q = N_{\text{hits}} + \frac{\chi^2_{\text{max}} - \chi^2}{\chi^2_{\text{max}} + 1},$$

where χ^2_{max} is the maximum acceptable χ^2 of tracks and N_{hits} is the number of assigned hits. Then duplicated tracks are removed, where duplicated means having half or more of their hits shared with another track with a higher quality.

Both of the track-finding approaches take advantage of an advanced unfolding of overlapped clusters [5]. The method exploits a so-called Maximum Likelihood - Expectation Maximization (MLEM or EM) deconvolution technique [6] (also known as Lucy-Richardson method or Bayesian unfolding). The essence of the method is that it iteratively solves the inverse problem of a distribution deconvolution. It is widely used in nuclear medicine for tomographic image reconstruction, in astronomy, and was also successfully tried for hit finding in silicon drift detectors. Effectively, this method improves the detector segmentation offering better conditions for making a decision about complex cluster splitting.

Under typical conditions, the track-finding efficiency of both of the methods is higher than 90%, with the Kalman filter approach being faster and giving a better resolution. As an example of the results of the track reconstruction in the forward muon spectrometer, Fig. 5.5 shows the reconstructed Υ invariant mass.

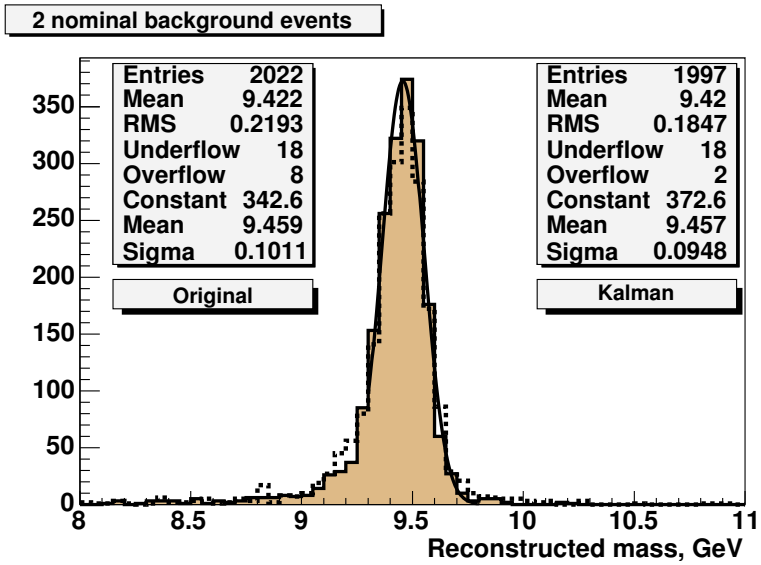

Figure 5.5: Reconstructed dimuon invariant mass in the region of the Υ mass.

5.4 Charged particle identification

Particle identification over a large momentum range and for many particle species is often one of the main requirements of high-energy physics experiments. The ALICE experiment is able to identify particles with momenta from 0.1 GeV/c to, in some cases, above 10 GeV/c. This can be achieved by combining several detecting systems that are efficient in narrower and complementary momentum sub-ranges. This combining is done following a Bayesian approach. The method is similar to the one described in Ref. [7] and satisfies the following requirements:

- It can combine the PID signals of different nature (e.g. dE/dx and time-of-flight measurements).
- When several detectors contribute to the PID, the procedure profits from this situation by providing an improved PID.
- When only some of the detectors identify a particle, the signals from the other detectors do not affect the combined PID.
- It takes into account the fact that the PID results depend on a particular track and event selection used in the analysis.

The PID procedure consists of three parts:

- The conditional probability density functions $r(s|i)$ to observe a PID response s from a particle of type i (the single-detector PID response functions) are obtained for all the detectors that provide the PID information. This is done by the calibration software using the Event Summary Data (ESD) for a subset of events as an input.
- For each reconstructed track, the global PID response $R(\bar{s}|i)$, which is a combination of the single detector response functions $r(s|i)$, is calculated taking into account possible effects of mismeasured PID signals. The results are written to the ESD and, later, are used in the physics analysis of the data. This is part of the reconstruction software.

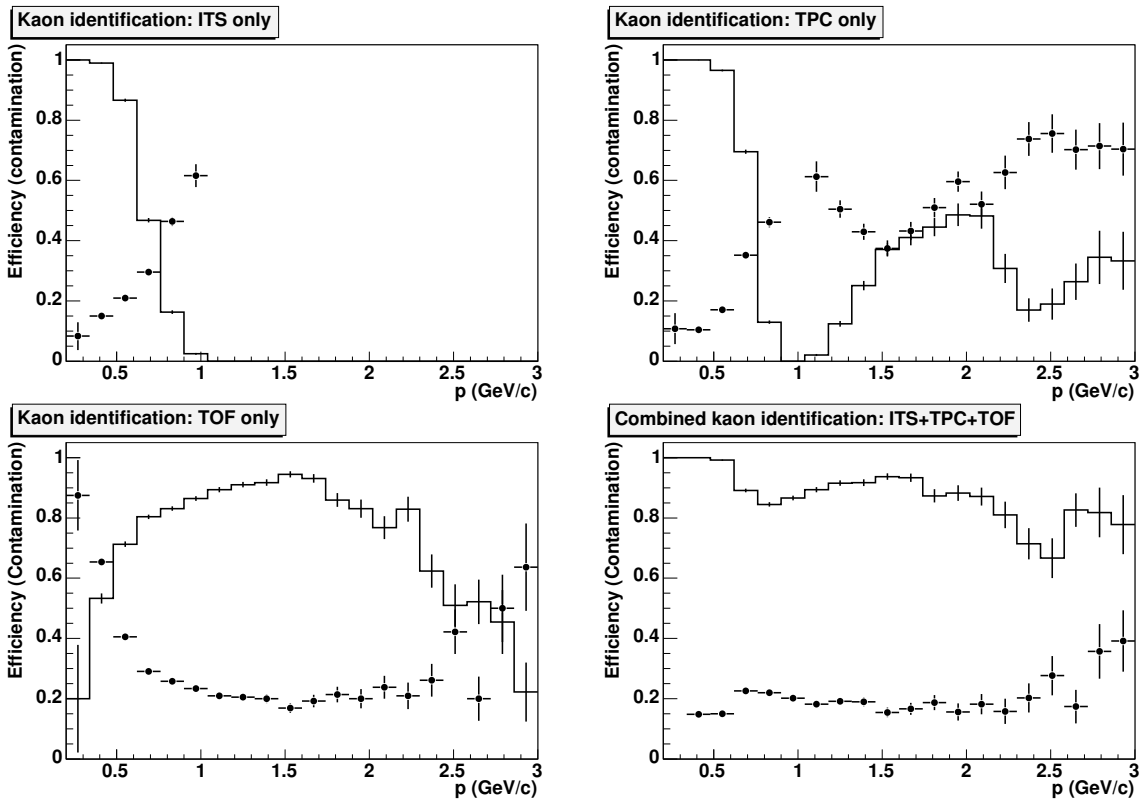

Figure 5.6: Single-detector efficiencies (solid line) and contaminations (points with error bars) of the charged kaon identification with the ITS, TPC and TOF stand-alone and the efficiency and contamination with all the detectors combined together.

- Finally, during a physics analysis, after the corresponding event and track selection is done, the *a priori* probabilities C_i for a track to be a particle of a certain type i within that selected track subset are estimated and the PID weights $W(i|\vec{s})$ are calculated by means of Bayes's formula:

$$W(i|\vec{s}) = \frac{R(\vec{s}|i)C_i}{\sum_{k=e,\mu,\pi,\ldots} R(\vec{s}|k)C_k}, \quad i = e,\mu,\pi,\ldots \tag{5.1}$$

  This part of the PID procedure belongs to the analysis software (see Chapter 5 of the ALICE Physics Performance Report Volume 2 [1] for the details).

The performance of identifying charged kaons in central HIJING Pb–Pb $\sqrt{s_{NN}} = 5.5$ TeV events using the ITS, TPC and the TOF as stand-alone detectors and the result for the combined PID are shown in Fig. 5.6. A track was considered as a charged kaon track, if the corresponding PID weight was the maximal. The PID efficiency is defined as the ratio of the number of correctly identified particles to the true number of particles of that type entering the PID procedure, and the contamination is the ratio of the number of mis-identified particles to the sum of correctly identified and mis-identified particles.

As can be seen in this figure, the efficiency and the contamination of the combined PID are significantly better, and less dependent on the momentum, than in the case of a single detector particle identification. The efficiency of the combined result is always higher than (or equal to) in the case of any of the detectors working stand-alone and the combined PID contamination is always lower than (or equal to) the contaminations obtained with the single detector PID.

The approach can easily adopt the PID information provided by the TRD. This improves the PID quality for all the particle types (by using the dE/dx measurements) and, in particular, for the electrons (by using the additional transition radiation signal) [8].

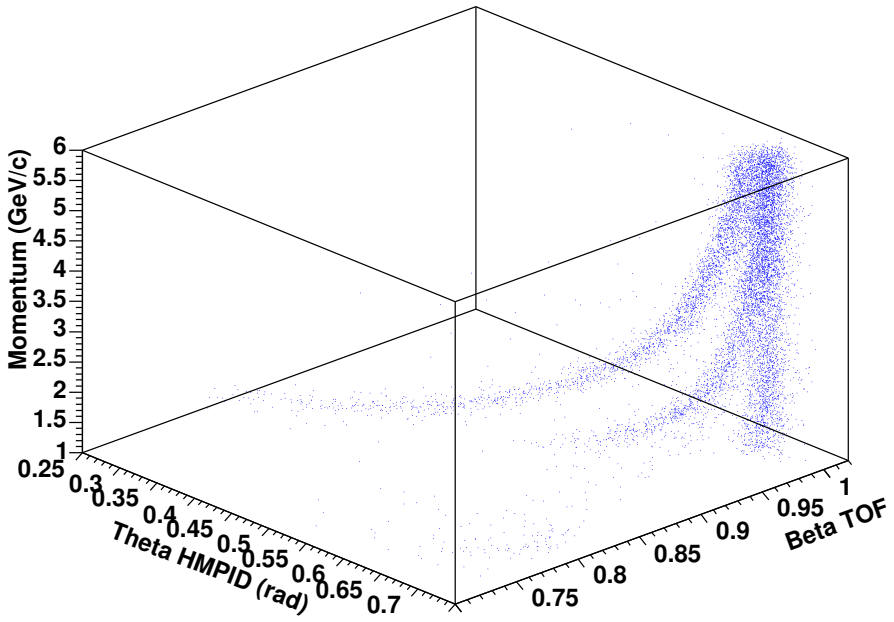

Figure 5.7: The correlation between the Cherenkov angle θ_c measured by HMPID and β measured by TOF for different particle species as a function of the particle momentum.

The identification of high momentum charged hadrons in the central rapidity region will be performed by the HMPID system, which consists of an array of seven RICH detectors. A combined identification of pions, kaons and protons between HMPID and TOF will be performed in momentum intervals partially overlapped, where a good PID for both the detectors could be achieved. The combined information from these detectors will improve the identification efficiency and will decrease significantly the contamination. The correlation between the reconstructed Cherenkov angle θ_c (measured by HMPID) and the β (measured by TOF) of pions, kaons and protons as a function of the momentum has been studied in simulated HIJING Pb–Pb central events. The results are shown in Fig. 5.7. The software to obtain the PID combined over HMPID and TOF is under development.

5.5 Photon and neutral meson reconstruction in the PHOS

Photons in ALICE are detected by the PHOton Spectrometer, PHOS [9], consisting of an electromagnetic calorimeter to measure 4-momenta of photons and a charged-particle veto detector (CPV) to discriminate neutral and charged particles. All the particles hitting the PHOS interact with the calorimeter medium producing showers, and deposit energy in its cells. The cells with signals are grouped into clusters that allow one to evaluate the total energy deposited by the particles in the calorimeter and the coordinate of the particle impacts on the surface of the detector. A special unfolding procedure is applied to split the clusters produced by particles with overlapping showers.

The particle identification in PHOS is based on three criteria [1]: time-of-flight measurement, shape of the showers produced by different particles in the calorimeter, and matching of the reconstructed point in the calorimeter and the reconstructed point in the CPV detector. The time of flight between the beam interaction time and the detection in PHOS is measured by the front-end electronics and allow the suppression of slow particles, especially low-energy ($E < 2$ GeV) nucleons. The shower shape produced by photons and electrons is different from the showers produced by hadrons. This difference is used to discriminate particles interacting with the calorimeter electromagnetically and hadronically. Matching of the charged particles with the calorimeter reconstructed point is provided by the CPV detector and allows the selection of neutral particles.

The direct photons are measured as an excess of the photon spectrum over the decay photons. To measure the decay photon spectrum, one reconstructs the spectra of neutral mesons decaying into photons (π^0, η, ω, etc). These spectra at low p_t are measured statistically via invariant-mass spectra of all photon combinations. In the high-multiplicity environment of heavy-ion collisions, the combinatorial background for the invariant-mass spectra is very high. As a demonstration, the spectrum of two-photon invariant mass at $1 < p_t < 1.5$ GeV/c in the most central Pb–Pb collisions at 5.5 TeV simulated by HIJING is shown in Fig. 5.8 (left). After the combinatorial background subtraction by the mixed-events

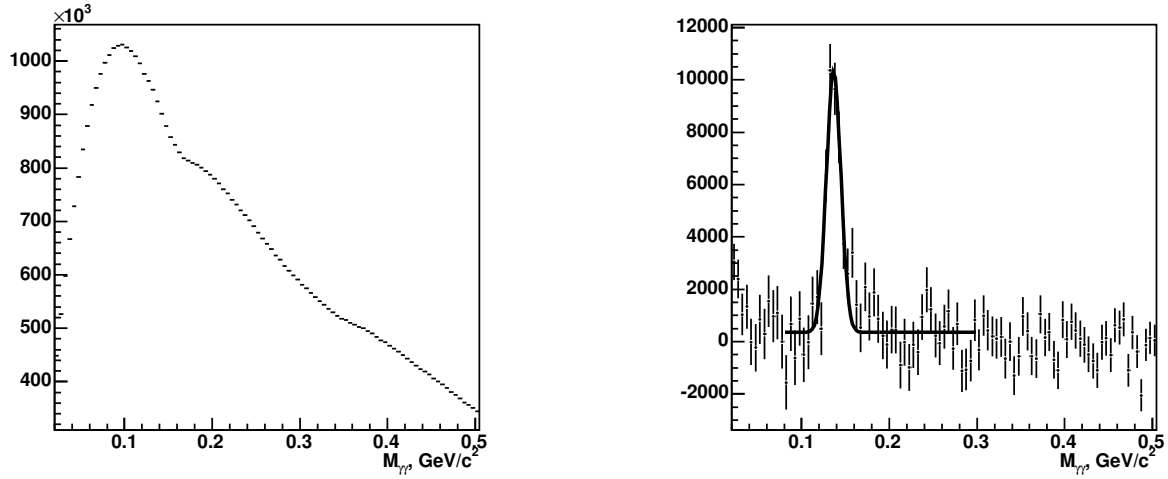

Figure 5.8: Two-photon invariant-mass distribution $1.0 < p_t < 1.5$ GeV/c in central Pb–Pb events (left), and after the combinatorial background subtraction (right).

technique, the invariant-mass spectrum clearly reveals the π^0 peak (Fig. 5.8, right). Having reconstructed spectra of all neutral mesons, one calculates the decay photon spectrum. The direct photon spectrum is obtained by subtracting the calculated decay photon spectrum from the total measured photon spectrum.

5.6 High-Level Trigger reconstruction

The algorithms in preparation for the High-Level Trigger (HLT) reconstruction code are implemented within the AliRoot reconstruction chain. This version is used to study the HLT track finding performance and replay the online trigger decisions during the offline data analysis. The code is organized as a 'virtual' detector reconstructor class which derives from the base `AliReconstructor` class and is called by the steering class `AliReconstruction`. The output of the HLT reconstruction is stored in an ESD object using the same format as the offline reconstruction. This facilitate the use of the offline analysis code and the comparison between offline and HLT reconstruction results.

So far the HLT reconstruction chain incorporates algorithms for fast track finding in the TPC and ITS which include:

- Cluster finder and track follower in the TPC [10, 11]. In the first step the cluster finder reconstructs the cluster centroids without using prior knowledge of the tracks. The resulting space points are then processed by the track finder, which forms track segments. Finally the space points are fitted to extract the track parameters.

- Hough transform track finding in the TPC [12]. The track finding is based on a Hough transform procedure applied to conformal mapped cluster boundaries. It combines a linear Hough transformation with fast filling of the Hough transform parameter space. The tracks are identified as peaks in the parameter space and the track parameters are provided by the peak centroids. The track-finding efficiency as a function of the event multiplicity is shown in Fig. 5.9.

- ITS clusterization, vertex reconstruction and track finding based on a version of the offline code optimized for time performance. Like the offline reconstruction approach, the TPC tracks from either the track follower or Hough transform are prolongated into the ITS. The track parameters and their covariance matrix at the point of closest approach to the primary vertex are stored in the ESD.

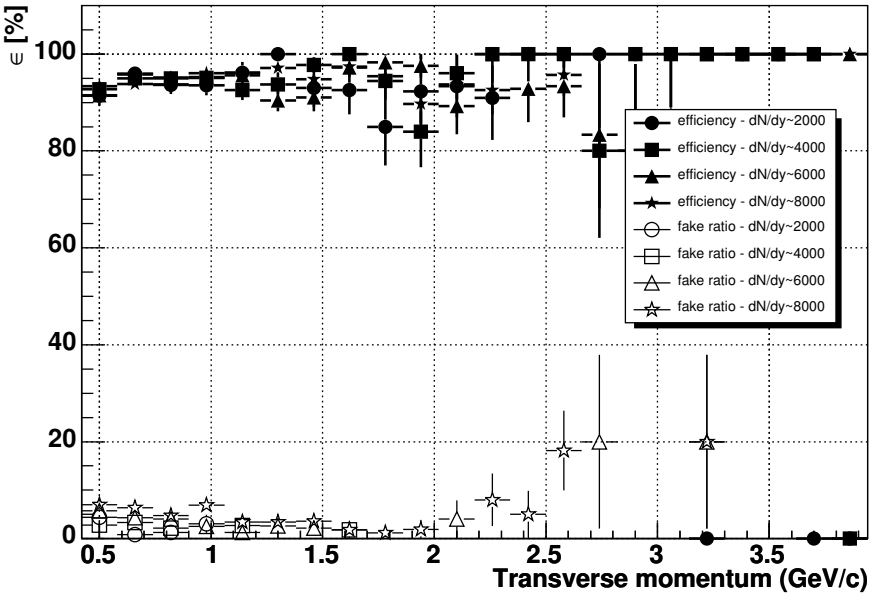

Figure 5.9: Hough transform track finding efficiency (closed symbols) and probability to obtain a fake track (open symbols) as a function of transverse momentum for different event multiplicities.

In order to implement the full reconstruction chain, it is foreseen to add Kalman-filter-based TRD track finding and PID as well as HLT muon spectrometer track reconstruction. The physics trigger algorithms which will actually be used by the HLT for online analysis of physics observables and selection of events and 'Regions of Interest' are under development. Within the AliRoot reconstruction chain they will be adapted to take as input the HLT ESD. The corresponding trigger decisions will be written in the same ESD object allowing for easy access during offline data analysis.

6 Data analysis

6.1 Introduction

The analysis of experimental data is the final stage of event processing and it is usually repeated many times. Analysis is a very diverse activity, where the goals of each particular analysis pass may differ significantly.

The ALICE detector is optimized for the reconstruction and analysis of heavy-ion collisions. In addition, ALICE has a broad physics programme devoted to pp and pA interactions.

The main points of the ALICE heavy-ion programme can be summarized as follows [1]:

- **global event characteristics:** particle multiplicity, centrality, energy density, nuclear stopping;
- **soft physics:** chemical composition (particle and resonance production, particle ratios and spectra, strangeness enhancement), reaction dynamics (transverse and elliptic flow, HBT correlations, event-by-event dynamical fluctuations);
- **hard probes:** jets, direct photons;
- **heavy flavours:** quarkonia, open charm and beauty production.

The pp and pA programme will provide, on the one hand, reference points for comparison with heavy ions. On the other hand, ALICE will also pursue genuine and detailed pp studies. Some quantities, in particular the global characteristics of interactions, will be measured during the first days of running exploiting the low-momentum measurement and particle identification capabilities of ALICE.

6.1.1 The analysis activity

We distinguish two main types of analysis: scheduled analysis and chaotic analysis. They differ in their data access pattern, in the storage and registration of the results, and in the frequency of changes in the analysis code. The detailed definition is given in Section 6.2.

In the ALICE Computing Model the analysis starts from the Event Summary Data (ESD). These are produced during the reconstruction step and contain all the information for the analysis. The size of the ESD is about one order of magnitude lower than the corresponding raw data. The analysis tasks produce Analysis Object Data (AOD) specific to a given set of physics objectives. Further passes for the specific analysis activity can be performed on the AODs, until the selection parameter or algorithms are changed.

A typical data analysis task usually requires processing of selected sets of events. The selection is based on the event topology and characteristics, and is done by querying the tag database (see Chapter 2). The tags represent physics quantities which characterize each run and event, and permit fast selection. They are created after the reconstruction and contain also the unique identifier of the ESD file. A typical query, when translated into natural language, could look like “Give me all the events with impact parameter in <range> containing jet candidates with energy larger than <threshold>”. This results in a list of events and file identifiers to be used in the consecutive event loop.

The next step of a typical analysis consists of a loop over all the events in the list and calculation of the physics quantities of interest. Usually, for each event, there is a set of embedded loops on the reconstructed entities such as tracks, V^0 candidates, neutral clusters, etc., the main goal of which is to select the signal candidates. Inside each loop a number of criteria (cuts) are applied to reject the background combinations and to select the signal ones. The cuts can be based on geometrical quantities

such as impact parameters of the tracks with respect to the primary vertex, distance between the cluster and the closest track, distance of closest approach between the tracks, angle between the momentum vector of the particle combination and the line connecting the production and decay vertices. They can also be based on kinematics quantities such as momentum ratios, minimal and maximal transverse momentum, angles in the rest frame of the combination. Particle identification criteria are also among the most common selection criteria.

The optimization of the selection criteria is one of the most important parts of the analysis. The goal is to maximize the signal-to-background ratio in case of search tasks, or another ratio (typically $\mathrm{Signal}/\sqrt{\mathrm{Signal}+\mathrm{Background}}$) in case of measurement of a given property. Usually, this optimization is performed using simulated events where the information from the particle generator is available.

After the optimization of the selection criteria, one has to take into account the combined acceptance of the detector. This is a complex, analysis-specific quantity which depends on the geometrical acceptance, the trigger efficiency, the decays of particles, the reconstruction efficiency, the efficiency of the particle identification and of the selection cuts. The components of the combined acceptance are usually parametrized and their product is used to unfold the experimental distributions or during the simulation of some model parameters.

The last part of the analysis usually involves quite complex mathematical treatments, and sophisticated statistical tools. Here one may include the correction for systematic effects, the estimation of statistical and systematic errors, etc.

6.2 Organization of the data analysis

The data analysis is coordinated by the Physics Board via the Physics Working Groups (PWGs). At present the following PWG have started their activity:

- detector performance;
- global event characteristics and soft physics (including proton–proton physics);
- hard probes: jets and direct photons;
- heavy flavours.

Scheduled analysis

The scheduled analysis typically uses all the available data from a given period, and stores and registers the results using Grid middleware. The tag database is updated accordingly. The AOD files, generated during the scheduled analysis, can be used by several subsequent analyses, or by a class of related physics tasks. The procedure of scheduled analysis is centralized and can be considered as data filtering. The requirements come from the PWGs and are prioritized by the Physics Board taking into account the available computing and storage resources. The analysis code is tested in advance and released before the beginning of the data processing.

Each PWG will require several sets of AOD per event, which are specific for one or a few analysis tasks. The creation of the AOD sets is managed centrally. The event list of each AOD set will be registered and the access to the AOD files will be granted to all ALICE collaborators. AOD files will be generated via Grid tools at different computing centres and will be stored on the corresponding storage elements. The processing of each file set will thus be done in a distributed way on the Grid. Some of the AOD sets may be quite small and would fit on a single storage element or even on one computer; in this case the corresponding tools for file replication, available in the ALICE Grid infrastructure, will be used.

Chaotic analysis

The chaotic analysis is focused on a single physics task and typically is based on the filtered data from the scheduled analysis. Each physicist also may access directly large parts of the ESD in order to search for rare events or processes. Usually the user develops the code using a small subsample of data, and changes the algorithms and criteria frequently. The analysis macros and software are tested many times on relatively small data volumes, both experimental and Monte Carlo. The output is often only a set of histograms. Such a tuning of the analysis code can be done on a local data set or on distributed data using Grid tools. The final version of the analysis will eventually be submitted to the Grid and will access large portions or even the totality of the ESDs. The results may be registered in the Grid file catalogue and used at later stages of the analysis. This activity may or may not be coordinated inside the PWGs, via the definition of priorities. The chaotic analysis is carried on within the computing resources of the physics groups.

6.3 Infrastructure tools for distributed analysis

6.3.1 gShell

The main infrastructure tools for distributed analysis have been described in Chapter 3. The actual middleware is hidden by an interface to the Grid, gShell [2], which provides a single working shell. The gShell package contains all the commands a user may need for file catalogue queries, creation of sub-directories in the user space, registration and removal of files, job submission and process monitoring. The actual Grid middleware is completely transparent to the user.

The gShell overcomes the scalability problem of direct client connections to databases. All clients connect to the gLite [3] API services. This service is implemented as a pool of preforked server daemons, which serve single-client requests. The client-server protocol implements a client state which is represented by a current working directory, a client session ID and time-dependent symmetric cipher on both ends to guarantee client privacy and security. The server daemons execute client calls with the identity of the connected client.

6.3.2 PROOF – the Parallel ROOT Facility

The Parallel ROOT Facility, PROOF [4] has been specially designed and developed to allow the analysis and mining of very large data sets, minimizing response time. It makes use of the inherent parallelism in event data and implements an architecture that optimizes I/O and CPU utilization in heterogeneous clusters with distributed storage. The system provides transparent and interactive access to terabyte-scale data sets. Being part of the ROOT framework, PROOF inherits the benefits of a performing object storage system and a wealth of statistical and visualization tools. The most important design features of PROOF are:

- transparency – no difference between a local ROOT and a remote parallel PROOF session;
- scalability – no implicit limitations on number of computers used in parallel;
- adaptability – the system is able to adapt to variations in the remote environment.

PROOF is based on a multi-tier architecture: the ROOT client session, the PROOF master server, optionally a number of PROOF sub-master servers, and the PROOF worker servers. The user connects from the ROOT session to a master server on a remote cluster and the master server creates sub-masters and worker servers on all the nodes in the cluster. All workers process queries in parallel and the results are presented to the user as coming from a single server.

PROOF can be run either in a purely interactive way, with the user remaining connected to the master and worker servers and the analysis results being returned to the user's ROOT session for further

analysis, or in an 'interactive batch' way where the user disconnects from the master and workers (see Fig. 3.4 on page 30). By reconnecting later to the master server the user can retrieve the analysis results for that particular query. This last mode is useful for relatively long running queries (several hours) or for submitting many queries at the same time. Both modes will be important for the analysis of ALICE data.

6.4 Analysis tools

This section is devoted to the existing analysis tools in ROOT and AliRoot. As discussed in the introduction, some very broad analysis tasks include the search for some rare events (in this case the physicist tries to maximize the signal-over-background ratio), or measurements where it is important to maximize the signal significance. The tools that provide possibilities to apply certain selection criteria and to find the interesting combinations within a given event are described below. Some of them are very general and are used in many different places, for example the statistical tools. Others are specific to a given analysis.

6.4.1 Statistical tools

Several commonly used statistical tools are available in ROOT [5]. ROOT provides classes for efficient data storage and access, such as trees and ntuples. The ESD information is organized in a tree, where each event is a separate entry. This allows a chain of the ESD files to be made and the elaborated selector mechanisms to be used in order to exploit the PROOF services. Inside each ESD object the data is stored in polymorphic containers filled with reconstructed tracks, neutral particles, etc. The tree classes permit easy navigation, selection, browsing, and visualization of the data in the branches.

ROOT also provides histogramming and fitting classes, which are used for the representation of all the one- and multi-dimensional distributions, and for extraction of their fitted parameters. ROOT provides an interface to powerful and robust minimization packages, which can be used directly during some special parts of the analysis. A special fitting class allows one to decompose an experimental histogram as a superposition of source histograms.

ROOT also has a set of sophisticated statistical analysis tools such as principal component analysis, robust estimator, and neural networks. The calculation of confidence levels is provided as well.

Additional statistical functions are included in `TMath`.

6.4.2 Calculations of kinematics variables

The main ROOT physics classes include 3-vectors and Lorentz vectors, and operations such as translation, rotation, and boost. The calculations of kinematics variables such as transverse and longitudinal momentum, rapidity, pseudorapidity, effective mass, and many others are provided as well.

6.4.3 Geometrical calculations

There are several classes which can be used for measurement of the primary vertex: `AliITSVertexerZ`, `AliITSVertexerPPZ`, `AliITSVertexerIons`, `AliITSVertexerTracks`. A fast estimation of the z-position can be done by `AliITSVertexerZ`, which works for both lead–lead and proton–proton collisions. An universal tool is provided by `AliITSVertexerTracks`, which calculates the position and covariance matrix of the primary vertex based on a set of tracks, and also estimates the χ^2 contribution of each track. An iterative procedure can be used to remove the secondary tracks and improve the precision.

Track propagation to the primary vertex (inward) is provided in AliESDtrack.

The secondary vertex reconstruction in case of V^0 is provided by `AliV0vertexer`, and in case of cascade hyperons by `AliCascadeVertexer`. An universal tool is `AliITSVertexerTracks`, which can

be used also to find secondary vertices close to the primary one, for example decays of open charm like $D^0 \to K^-\pi^+$ or $D^+ \to K^-\pi^+\pi^+$. All the vertex reconstruction classes also calculate distance of closest approach (DCA) between the track and the vertex.

The calculation of impact parameters with respect to the primary vertex is done during the reconstruction and the information is available in `AliESDtrack`. It is then possible to recalculate the impact parameter during the ESD analysis, after an improved determination of the primary vertex position using reconstructed ESD tracks.

6.4.4 Global event characteristics

The impact parameter of the interaction and the number of participants are estimated from the energy measurements in the ZDC. In addition, the information from the FMD, PMD, and T0 detectors is available. It gives a valuable estimate of the event multiplicity at high rapidities and permits global event characterization. Together with the ZDC information it improves the determination of the impact parameter, number of participants, and number of binary collisions.

The event plane orientation is calculated by the `AliFlowAnalysis` class.

6.4.5 Comparison between reconstructed and simulated parameters

The comparison between the reconstructed and simulated parameters is an important part of the analysis. It is the only way to estimate the precision of the reconstruction. Several example macros exist in AliRoot and can be used for this purpose: `AliTPCComparison.C`, `AliITSComparisonV2.C`, etc. As a first step in each of these macros the list of so-called 'good tracks' is built. The definition of a good track is explained in detail in the ITS [6] and TPC [7] Technical Design Reports. The essential point is that the track goes through the detector and can be reconstructed. Using the 'good tracks' one then estimates the efficiency of the reconstruction and the resolution.

Another example is specific to the MUON arm: the `MUONRecoCheck.C` macro compares the reconstructed muon tracks with the simulated ones.

There is also the possibility to calculate directly the resolutions without additional requirements on the initial track. One can use the so-called track label and retrieve the corresponding simulated particle directly from the particle stack (`AliStack`).

6.4.6 Event mixing

One particular analysis approach in heavy-ion physics is the estimation of the combinatorial background using event mixing. Part of the information (for example the positive tracks) is taken from one event, another part (for example the negative tracks) is taken from a different, but 'similar' event. The event 'similarity' is very important, because only in this case the combinations produced from different events represent the combinatorial background. Typically 'similar' in the example above means with the same multiplicity of negative tracks. One may require in addition similar impact parameters of the interactions, rotation of the tracks of the second event to adjust the event plane, etc. The possibility for event mixing is provided in AliRoot by the fact that the ESD is stored in trees and one can chain and access simultaneously many ESD objects. Then the first pass would be to order the events according to the desired criterion of 'similarity' and to use the obtained index for accessing the 'similar' events in the embedded analysis loops. An example of event mixing is shown in Fig. 6.1. The background distribution has been obtained using 'mixed events'. The signal distribution has been taken directly from the Monte Carlo simulation. The 'experimental distribution' has been produced by the analysis macro and decomposed as a superposition of the signal and background histograms.

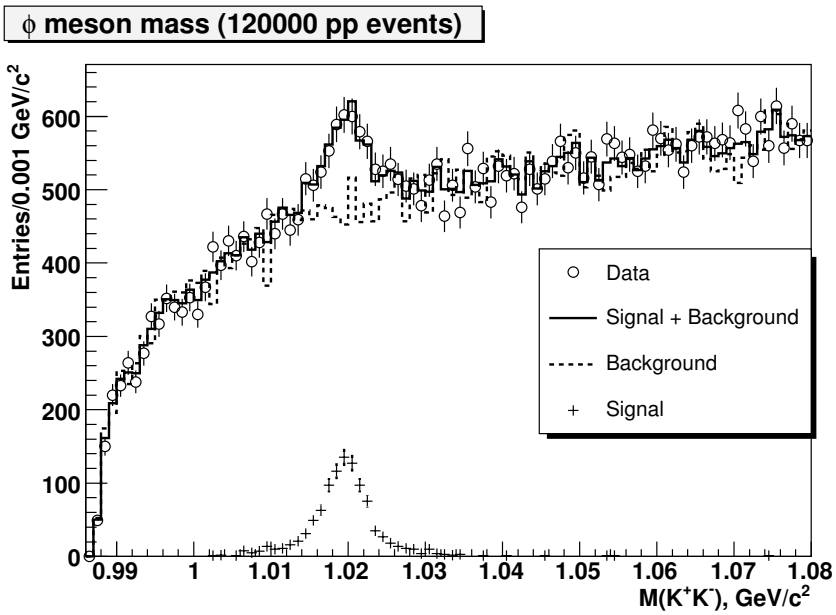

Figure 6.1: Mass spectrum of the ϕ meson candidates produced inclusively in the proton–proton interactions.

6.4.7 Analysis of the High-Level Trigger (HLT) data

This is a specific analysis which is needed in order to adjust the cuts in the HLT code, or to estimate the HLT efficiency and resolution. AliRoot provides a transparent way of doing such an analysis, since the HLT information is stored in the form of ESD objects in a parallel tree. This also helps in the monitoring and visualization of the results of the HLT algorithms.

6.4.8 Visualization

The visualization classes give the possibility for prompt inspection of the simulation and reconstruction results. The initial version of the visualization is available in the `AliDisplay` class. Another more elaborated module `DISPLAY` is under development.

6.5 Existing analysis examples in AliRoot

There are several dedicated analysis tools available in AliRoot. Their results were used in the Physics Performance Report and described in ALICE internal notes. There are two main classes of analysis: the first one based directly on ESD, and the second one extracting first AOD, and then analysing it.

- **ESD analysis**

  V^0 and cascade reconstruction/analysis

  The V^0 candidates are reconstructed during the combined barrel tracking and stored in the ESD object. The following criteria are used for the selection: minimal-allowed impact parameter (in the transverse plane) for each track; maximal-allowed DCA between the two tracks; maximal-allowed cosine of the V^0 pointing angle (angle between the momentum vector of the particle combination and the line connecting the production and decay vertices); minimal and maximal radius of the fiducial volume; maximal-allowed χ^2. The last criterion requires the covariance matrix of track parameters, which is available only in `AliESDtrack`. The reconstruction is performed by `AliV0vertexer`. This class can be used also in the analysis. An example of reconstructed kaons taken directly from the ESDs is shown in Fig.6.2.

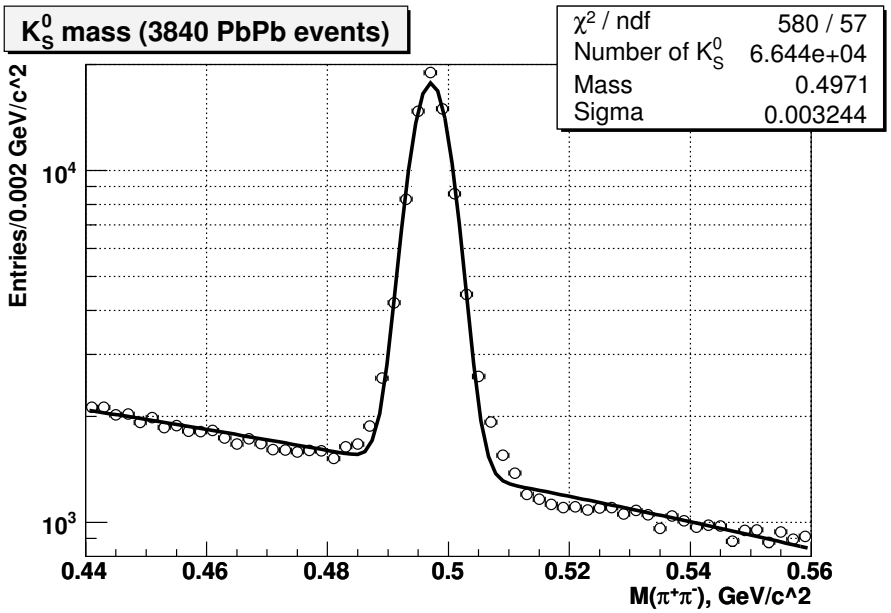

Figure 6.2: Mass spectrum of the K^0_S meson candidates produced inclusively in the Pb–Pb collisions.

The cascade hyperons are reconstructed using the V^0 candidate and 'bachelor' track selected according to the cuts above. In addition, one requires that the reconstructed V^0 effective mass belongs to a certain interval centred in the true value. The reconstruction is performed by `AliCascadeVertexer`, and this class can be used in the analysis.

Open charm

This is the second elaborated example of ESD analysis. There are two classes, `AliD0toKpi` and `AliD0toKpiAnalysis`, which contain the corresponding analysis code. The decay under investigation is $D^0 \to K^-\pi^+$ and its charge conjugate. Each D^0 candidate is formed by a positive and a negative track, selected to fulfil the following requirements: minimal-allowed track transverse momentum, minimal-allowed track impact parameter in the transverse plane with respect to the primary vertex. The selection criteria for each combination include maximal-allowed distance of closest approach between the two tracks, decay angle of the kaon in the D^0 rest frame in a given region, product of the impact parameters of the two tracks larger than a given value, pointing angle between the D^0 momentum and flight-line smaller than a given value. The particle identification probabilities are used to reject the wrong combinations, namely (K, K) and (π, π), and to enhance the signal-to-background ratio at low momentum by requiring the kaon identification. All proton-tagged tracks are excluded before the analysis loop on track pairs. More details can be found in Ref. [8].

Quarkonia analysis

Muon tracks stored in the ESD can be analysed by the macro `MUONmassPlot_ESD.C`. This macro performs an invariant-mass analysis of muon unlike-sign pairs and calculates the combinatorial background. Quarkonia $p_t$and rapidity distribution are built for J/ψ and Υ. This macro also performs a fast single-muon analysis: p_t, rapidity, and θ vs φ acceptance distributions for positive and negative muon tracks with a maximal-allowed χ^2.

- **AOD analysis**

  Often only a small subset of information contained in the ESD is needed to perform an analysis. This information can be extracted and stored in the AOD format in order to reduce the computing resources needed for the analysis.

The AOD analysis framework implements a set of tools like data readers, converters, cuts, and other utility classes. The design is based on two main requirements: flexibility and common AOD particle interface. This guarantees that several analyses can be done in sequence within the same computing session.

In order to fulfil the first requirement, the analysis is driven by the 'analysis manager' class and particular analyses are added to it. It performs the loop over events, which are delivered by an user-specified reader. This design allows the analyses to be ordered appropriately if some of them depend on the results of the others.

The cuts are designed to provide high flexibility and performance. A two-level architecture has been adopted for all the cuts (particle, pair and event). A class representing a cut has a list of 'base cuts'. Each base cut implements a cut on a single property or performs a logical operation (and, or) on the result of other base cuts.

A class representing a pair of particles buffers all the results, so they can be re-used if required.

Particle momentum correlations (HBT) – HBTAN module

Particle momentum correlation analysis is based on the event-mixing technique. It allows one to extract the signal by dividing the appropriate particle spectra coming from the original events by those from the mixed events.

Two analysis objects are currently implemented to perform the mixing: the standard one and the one implementing the Stavinsky algorithm [9]. Others can easily be added if needed.

An extensive hierarchy of the function base classes has been implemented facilitating the creation of new functions. A wide set of the correlation, distribution and monitoring functions is already available in the module. See Ref. [10] for the details.

The package contains two implementations of weighting algorithms, used for correlation simulations (the first developed by Lednicky [11] and the second due to CRAB [12]), both based on an uniform interface.

Jet analysis

The jet analysis [13] is available in the module JETAN. It has a set of readers of the form `AliJetParticlesReader<XXX>`, where `XXX = ESD`, `HLT`, `KineGoodTPC`, `Kine`, derived from the base class `AliJetParticlesReader`. These provide an uniform interface to the information from the kinematics tree, from HLT, and from the ESD. The first step in the analysis is the creation of an AOD object: a tree containing objects of type `AliJetEventParticles`. The particles are selected using a cut on the minimal-allowed transverse momentum. The second analysis step consists of jet finding. Several algorithms are available in the classes of the type `Ali<XXX>JetFinder`. An example of AOD creation is provided in the `createEvents.C` macro. The usage of jet finders is illustrated in `findJets.C` macro.

V^0 AODs

The AODs for V^0 analysis contain several additional parameters, calculated and stored for fast access. The methods of the class `AliAODv0` provide access to all the geometrical and kinematics parameters of a V^0 candidate, and to the ESD information used for the calculations.

MUON

There is also a prototype MUON analysis provided in `AliMuonAnalysis`. It simply fills several histograms, namely the transverse momentum and rapidity for positive and negative muons, the invariant mass of the muon pair, etc.

The analysis framework is one of the most active fields of development. Many new classes and macros are in preparation. Some of them are already tested on the data produced during the Physics Data Challenge 2004 and will become part of the ALICE software.

7 Computing model and capacity requirements

7.1 Introduction

This chapter describes the computing model for the processing of the data produced by the ALICE detector in pp and A–A collisions every year. The computing model of the LHC experiements has been already reviewed in 2001 during the so-called LHC Computing Review (LHCCR) [1]. The model presented in this document however contains several changes with respect to the 2001 version, and it has been reviewed by a subsequent LHCC review that focussed mainly on the requested resources [2]. Compared to the 2001 version, the model has been further developed and refined thanks to the experience gained during the Physics Data Challenges. The amount of permanent storage has increased because a duplication of raw data at external Tier 1s is now foreseen in the model. Disk storage has substantially increased: the new estimate is deduced from the disk usage made during the Physics Data Challenges. This document contains also an evaluation of the contributions of Tier 2 centres, which was not present in the ALICE LHCC estimates.

7.2 Input parameters

The input parameters for the present computing model are derived from information contained in the ALICE Trigger, DAQ, HLT and Control System TDR [3] and the ALICE Physics Performance Report Vol. 1 [4]. All input parameters are to be considered as our best estimates at the moment. They are based on the nominal figures of a standard data-taking year. The resources requirements during the commissioning of the machine and the two first years of running have been estimated as a percentage of a standard data-taking year, taking into account the information available at this moment about the LHC machine schedule.

Independently from the LHC pp luminosity, the beam parameters will be adjusted to achieve a luminosity of at most 3×10^{30} cm^{-2}s^{-1} leading to an interaction of 200 kHz and an average pileup of about 36 events during the TPC drift time (88 μs). The parameters adopted for Pb–Pb collisions are listed in Table 7.1.

Table 7.1: Parameters adopted for Pb–Pb collisions.

Centre-of-mass energy	5.5A TeV
Luminosity	5×10^{25} cm^{-2}s^{-1} in 2008
	5×10^{26} cm^{-2}s^{-1} (average luminosity) from 2009 onward
Total reaction cross-section	8 b
Nominal collision rate	4×10^{3} Hz Pb–Pb collisions at average luminosity

The trigger rate and RAW data bandwidth are independent of the luminosity as trigger thresholds and conditions will be adjusted to saturate the bandwidth. During pp collisions, more than one bunch crossing will happen during the gating of the TPC on account of its long drift time, and therefore more than one collision may be recorded (pileup). The amount of this pileup depends on luminosity, and therefore the event size will increase with the luminosity. Studies are under way to remove these pileup events on line with the HLT system. However, given the criticality of this operation, we do not foresee doing it in the first years of pp running.

During the first year of operation, the following running scenario has been assumed. The first pp run will take place over a reduced time, from the beginning of July 2007 until the end of September 2007. Assuming the maximum luminosity allowed for ALICE, this first run will deliver 40% of the pp data during a standard data-taking year. This run will be followed by a short Pb–Pb pilot run with reduced luminosity which will deliver at most 20% of the Pb–Pb data we will collect in one standard data-taking year. The first Pb–Pb run with a still reduced luminosity will occur at the end of 2008, the second year of LHC operation. As we have explained this will not imply a reduction in the data rate, as we will adjust the trigger accordingly. These assumptions have been taken into account to establish the rampup of resources deployment required by ALICE.

Below we shall discuss the following types of data:

- **RAW**: raw data as recorded by the DAQ (real) or simulated (Monte Carlo, MC);
- **ESD**: Event Summary Data as produced by the reconstruction program;
- **AOD**: Physics Analysis Object Data derived from the ESD and similar to today's n-tuples;
- **TAG**: Event tags for event selection.

As far as the basic RAW data parameters are concerned, for both pp and Pb–Pb an average acquisition rate is considered. During a pp run, the rate can go up to 500 Hz, however, this is in special cases and for a short period of time only. Since for Pb–Pb collisions the event size changes significantly with the centrality of the collision, a normalized rate, taking an average event size of 12.5 MB, is considered. This event size has been calculated assuming a charged-particle density of $\mathrm{d}N/\mathrm{d}y = 4000$. The actual trigger rate will be the sum of several event types with different sizes and rates.

The data taking parameters are summarised in Table 7.2.

Table 7.2: Data-taking parameters.

	pp	Pb–Pb
Event recording rate (Hz)	100	100
Event recording bandwidth (MB/s)	100	1250
Running time per year (Ms)	10	1
Events per year	10^9	10^8

7.3 The computing model

The computing model is subject to change with time. During the first years of running, even if luminosities might be below the nominal luminosity, only slightly less data will be recorded than during runs with nominal luminosities. The reason is that selective triggers will become operational only after a necessary training period needed to understand the various detection systems. Data from this early period will be processed offline rather differently than data from later runs. Understanding the detectors and training of the alignment, calibration, reconstruction and analysis algorithms will required many processing passes by many physicists on a subset of raw data. This phase must be limited in time and followed by an early fast processing of the total set of available data to guarantee early access to physics results. This fast processing must, however, be complete to preserve the richness of the data and the discovery potential in the early run. Therefore, adequate computing must already be available when the LHC provides the first pp collisions.

In the following we discuss an offline processing scenario, from which the required computing resources are estimated, and which we foresee to apply during standard data-taking years at nominal luminosities. We assume that a normal run period is split into:

- seven months (effective 10^7 s) of pp collisions,
- one month (effective 10^6 s) of heavy-ion collisions, and
- four months of winter shutdown.

The overall organized processing (calibration, alignment, reconstruction and scheduled analysis) is scheduled and prioritized in the ALICE Physics Working Groups (PWG) and by the ALICE Physics Board. The end-user analysis (sometimes called 'chaotic analysis') is performed in the context of the PWGs and, in case of lack of resources, it is prioritized by the Physics Board.

Depending on the middleware that will be in use at the time of LHC start-up, a more or less hierarchical organization of computing resources (Tier 0, Tier 1, Tier 2) will be in order. Although ALICE believes that most likely the democratic cloud model rather than the strict hierarchy of the MONARC model will prevail, the estimate of the required resources uses the concepts and names introduced by MONARC (see also Section 3.2).

- Tier 0 provides permanent storage for the raw data, distributes raw data to Tier 1 and performs the calibration and alignment task and the first reconstruction pass. Our model foresees that one Tier 1 and one Tier 2 be co-located with the Tier 0 at CERN.
- Tier 1s outside CERN provide permanent storage of a copy of the raw data. All Tier 1s perform the subsequent reconstruction passes, the scheduled analysis tasks, and the reconstruction of Pb–Pb MC data; they have the responsibility for the long term storage of data processed at Tier 1s and Tier 2s.
- Tier 2s generate and reconstruct the simulated Monte Carlo data and perform the chaotic analysis.
- Tier 0, Tier 1s and Tier 2s provide short-term storage with fast access to multiple copies of active data, raw (only a fraction) and processed.

Although the scenario may change to adjust to the available computing resources and Grid middleware functionality at a given time, or to provide rapid feedback to physics results obtained during the initial reconstruction and analysis, it is unlikely that the estimated needs of computing resources will vary considerably. Changes of the order of up to 50% in processing, short-term storage, or long-term storage must however be anticipated. Unexpected features in the operation of the Grid resources, which might result in efficiencies lower than anticipated, and which could not be predicted during the Data Challenges, could also require more or less important changes in the needed computing resources.

To estimate the computing resources required by the present model, the efficiency factors for the usage of processors (CPU), short-term storage (disk), and long-term mass storage (MS) listed in Table 7.3 have been adopted. These are factors agreed by the LCG project and used by all the LHC experiments.

7.4 CPU requirements

7.4.1 Parameter values

The ALICE Data Acquisition system will record pp and heavy-ion data at a rate of 100 Hz during the previously discussed effective time of 10^7 s and 10^6 s, respectively (see Table 7.2). The raw data size will vary with the trigger selection. The product trigger-rate times raw-data-size is limited by the maximum bandwidth available to transfer data to the mass storage system, 1.25 GB/s.

Table 7.3: Efficiency factors for the use of processors (CPU), short-term storage (disk), and long-term mass storage (MS).

Efficiency factors (%)	
Efficiency for scheduled CPU	85
Efficiency for chaotic CPU	60
Disk utilization efficiency	70
MS utilization efficiency	100

The processing power required for the reconstruction of data, raw as well as Monte Carlo, has been estimated from the performance of the reconstruction algorithms presently in use and tested on Monte Carlo data. We have taken as a value 80% of the present reconstruction times, which takes into account the optimization of the code and its increase in complexity within a reasonable safety factor. For Pb–Pb collisions an average charged-particle multiplicity of 4000 has been adopted. This is a conservative estimation based on the recent results from RHIC and the theoretical extrapolations at LHC energy. The real value might, however, be smaller as suggested by the RHIC data. The reconstruction algorithms are not yet entirely optimized, and they are not complete as they do not yet include calibration or alignment procedures. The figures quoted for reconstruction try to take these factors into account, considering the evolution of the code performance during the past years.

During the first years there will be a full first reconstruction pass followed by a 'second pass' which will in reality be composed of several short runs to tune the reconstruction programs based on the results of the first full pass. It is expected that the resources needed by this activity will correspond to a full reconstruction pass. Then we will run a second full reconstruction pass on all data of that year's run with final condition information. As time passes, we expect to improve our capability to derive good condition information soon after the first reconstruction pass. However, we also expect the need to reconstruct and analyse data coming from previous runs. So the estimation of three reconstruction passes is reasonable over the foreseeable lifetime of the experiment, if compound with the foreseen upgrade of the computing equipment (see Section 7.7).

The computing power required for analysis has been estimated based on a limited variety of Grid-enabled analysis use-cases performed during the Physics Data Challenges. Since the analysis algorithms will vary in complexity depending on the physics channel under study, the computing power for analysis will have to be revised once a larger set of analysis classes has been exercised on the Grid. A variation of a factor up to two must be anticipated.

The computing power required for the generation of Monte Carlo data is better under control. It will be subject to changes only if the real charged-particle multiplicity differs substantially from the predicted by the HIJING Monte Carlo generator. Table 7.4 lists the values adopted for the various parameters entering our estimation. The CPU power quoted for simulation includes event generation, tracking of particles through the detectors (an average value for the GEANT 3 and FLUKA transport codes has been adopted), and digitization of the generated signals.

Chaotic analysis tasks are well-focused analyses performed by single users on a subset of the data, possibly residing entirely at given Tier 2 centres. Scheduled analyses tasks gather many analysis from users and thus require much more CPU power than single-analysis tasks. They are performed on the entire set of reconstructed data, once per reconstruction pass. To optimize data transfer, scheduled analysis is best performed at Tier 1 centres.

Table 7.4: Computing power required for the processing (simulation includes event generation, tracking and digitization) of the ALICE real and Monte Carlo data.

Processing power parameters			
		pp	HI
Reconstruction	KSI2k×s/event	5.9	740.0
Chaotic analysis	KSI2k×s/event	0.6	8.3
Scheduled analysis	KSI2k×s/event	16.0	240.0
Simulation	KSI2k×s/event	39.0	17000.0
Reconstruction passes	–	3	3
Chaotic analysis passes	–	20	20
Scheduled analysis passes	–	3	3

7.4.2 Raw-data processing strategy

The scheduled processing of raw data consists of the reconstruction of the raw data and the creation of ESD objects.

Although the processing strategies vary between pp and heavy-ion runs, they have in common that the first reconstruction pass must be fast and must start immediately after (for heavy-ion) or during (for pp) data taking to allow for rapid discoveries and to establish quickly the overall properties of the collisions. On the other hand, the first heavy-ion reconstruction pass must be finished well before the start of the next heavy-ion run (at least six months before) to be able to define from the data analysis the running conditions. For heavy-ion runs, this first reconstruction is preceded by a computing-intensive calibration and alignment task not exceeding one month in duration and taking place during heavy-ion data acquisition. The additional reconstruction passes will consist of a full-fledged reconstruction including fine tuning of all the parameters of the algorithms.

Calibration and alignment tasks are performed quasi online on the events stored on a disk buffer at Tier 0. The size of this buffer is the equivalent of two days of heavy-ion data taking, i.e. 200 TB.

7.4.2.1 pp data processing

The data collected during pp runs, being less demanding in terms of computing resources than heavy-ion data for their reconstruction, will be processed online or quasi online during data taking (seven months according to our standard data-taking year) at the CERN Tier 0. The data temporarily stored on the Tier 0 disk buffer (one day of Pb–Pb running is the equivalent of 10 days pp data taking) are processed, successively calibrated, and reconstructed. Reconstructed data will be kept for long-term storage locally at Tier 0 and distributed to the Tier 1s. The second and the third reconstruction passes will be distributed in Tier 1s and spread over a period of six months.

7.4.2.2 Heavy-ion data processing

Because of the much larger data recording rate in heavy-ion mode, online reconstruction of the entire set of data would require unaffordable computing resources. The first reconstruction pass will therefore last four months and can start immediately after the data taking ends, or even during the run depending on the performance of the calibration and alignment task. It is crucial that this first reconstruction ends well before the next heavy-ion run starts in order to allow for enough time to analyse the data and elaborate the new running conditions. We estimate that the time required for this is of the order of six months. The second and third reconstruction passes will be distributed in Tier 1s and can be done over a period

of six months. Obviously, in such a scenario, pp and A–A reconstruction passes will be concurrent. Two different reconstruction tasks, one for pp and one for A–A, will be permanently active in Tier 1s, and at Tier 0 either the first pp or the first A–A reconstruction will be active (see the sketch of a possible processing scenario in Fig. 7.1).

The resources required to perform the ALICE pp and heavy-ion data reconstruction within such a scenario are listed in Table 7.5.

Table 7.5: Annual processing resources required to reconstruct and analyse the real pp and A–A data and to process (generation, reconstruction, and analysis) Monte Carlo data.

	CPU (MSI2K)		
	Integrated	Average	Max.
MC (simulation+reconstruction)	150	12.0	12.0
Real reconstruction	94	7.8	13.0
Scheduled analysis	96	8.0	8.4
Chaotic analysis	16	1.3	1.3
Alignment & calibration	3	0.3	3.0

7.4.3 Monte Carlo data simulation

The production scheme of Monte Carlo differs for pp and heavy-ion simulations. In the case of pp simulations, Monte Carlo data will be generated in an amount similar to that of collected real data (10^9 events per year). To avoid requesting a prohibitive amount of resources and also to enrich generated events with physics signal of interest, we have adopted a merging technique for heavy-ion simulation. Full-fledged, heavy-ion events are generated in a limited amount (10^7 events per year) in sets of different impact parameters (minimum-bias, peripheral, and central); these events are called the 'underlying events'. The different signal events are then generated, merged into one underlying event, at the digits level, and the merged event is reconstructed in a single task. In this way, the underlying events can be reused several times. From the data challenges experience we have fixed to 10 the number of times an underlying event can be reused and preserve realistic event-to-event fluctuations.

The Monte Carlo pp and heavy-ion events are typically generated and reconstructed at the Tier 2 centres.

The annual resources required to process the Monte Carlo data (generation, reconstruction) are listed in Table 7.5.

7.4.4 Data analysis

Based on the experience of past experiments, the data analysis will consume a large fraction of the total amount of resources. At variance with the reconstruction and the simulation tasks, not all the analysis tasks can be easily scheduled as they will involve potentially all the physicists of the collaboration applying customized algorithms many times on subsets of the data of various sizes. This way of performing analysis is called chaotic analysis. However, it is foreseen to perform scheduled analysis during the same production tasks as the reconstruction passes and therefore over the entire set of data. This scheduled analysis will group in each pass all the official algorithms approved by the Physics Board. The time needed to analyse reconstructed events depends very much on the complexity of the analysis algorithm and on the overhead introduced by the Grid processing. The latter can only be predicted once the analysis on the Grid has been exercised with the final middleware and at the final scale. The processing power we have considered is an average value which can vary largely from one analysis to the other. To estimate

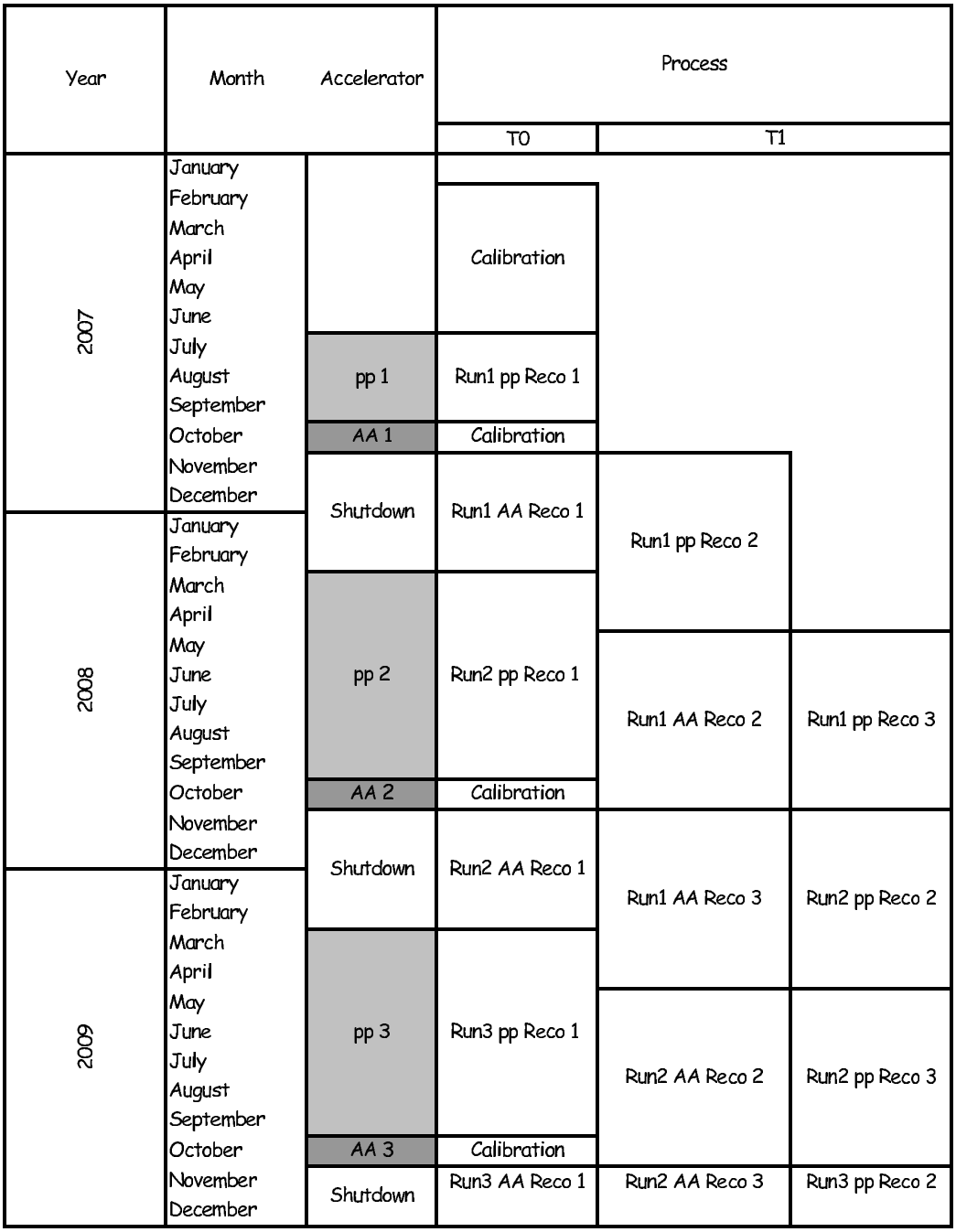

Figure 7.1: Processing scenario used to estimate the resources required at Tier 0, Tier 1s and Tier 2s.

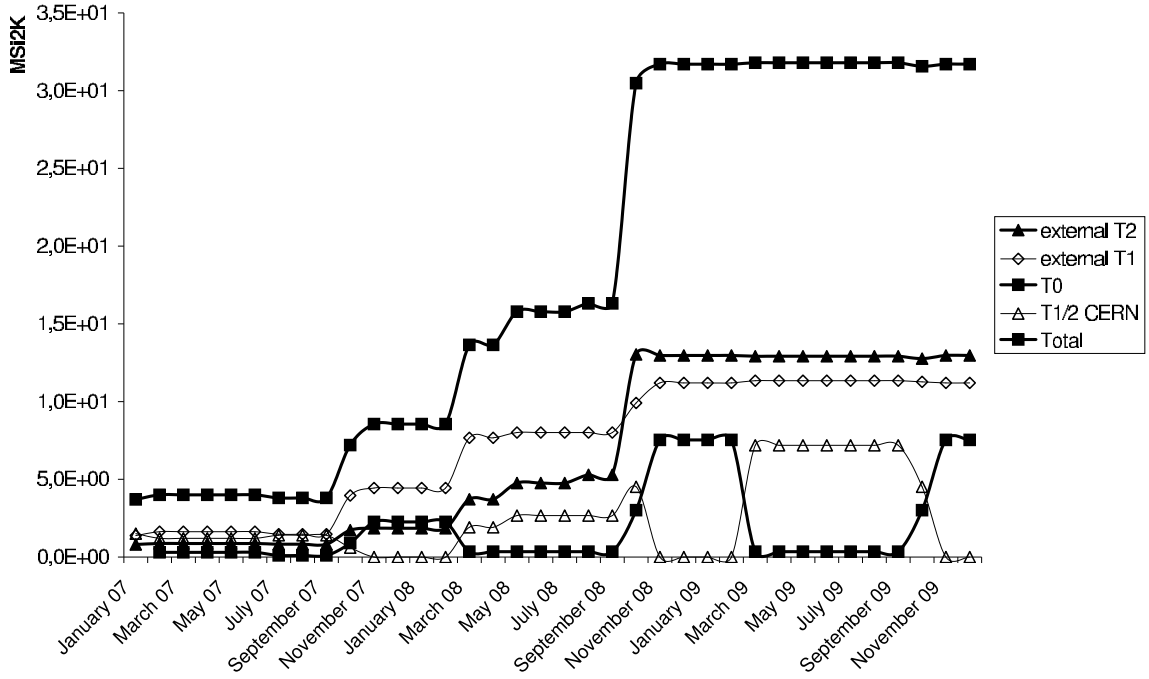

Figure 7.2: Profile of processing resources required at Tier 0, Tier 1s, and Tier 2s to process all ALICE data.

the processing resources required for the analysis we have considered that 200 physicists will exercise their algorithms many times on a small fraction of the data and produce physics results on a larger subset of the data. We furthermore assumed that the various analyses launched by the physics working groups are performed only once over the entire set of reconstructed data. Scheduled analysis, regrouping many analysis tasks, will require a larger amount of computing power per event than chaotic analysis. Individuals will also perform analyses using computing resources available at their home institutes. These resources are not accounted for in the present evaluation of the requirements.

The annual required resources to analyse real data are listed in Table 7.5.

From the scenario sketched in Fig. 7.1 on the preceding page, we deduce in Fig. 7.2 the profile of CPU resources required, including the ramp-up scenario.

The final values of resources required during a standard year of running at Tier 0, Tier 1s and Tier 2s are summarized in Table 7.8 on page 71.

7.5 Storage requirements

All data produced by the ALICE detector will be stored permanently for the duration of the experiment. This includes the raw data, a second copy of the raw data, one set of Monte Carlo data, reconstructed data from all reconstruction passes, analysis objects, calibration and condition data. A fraction of the produced data will be kept on short-term storage, providing rapid access to data frequently processed and for I/O dominated tasks. The media for permanent and transient storage will be decided according to the technology available. Currently the permanent storage medium is magnetic tape and the transient storage medium is disk. The ratio between disk and tape storage will change with time and will be dictated by the price–performance ratio. The parameters used to estimate the storage requirements are derived from the Data Challenge experience and are reported in Table 7.2 and Table 7.5.

7.5.1 Permanent storage

Raw data from pp and heavy-ion runs are copied onto permanent storage at Tier 0. They are further exported to the external Tier 1s, each one keeping a fraction of raw data on permanent storage, thus

Table 7.6: Size per event and per year of raw data produced by the ALICE experiment in pp and in Pb–Pb runs and by the processing of raw and Monte Carlo data.

	Real data (MB)				Monte Carlo data (MB)	
	Raw	ESD	AOD	Event catalogue	Raw	ESD
pp per event	1.0[a]	0.04	0.004	0.01	0.4	0.04
per year ($\times 10^9$)	1.1	0.18			0.4	0.1
Heavy-ion per event	12.5	2.5	0.25	0.01	300	2.5
per year ($\times 10^9$)	1.4	1.0			3.0	0.9

[a] It is assumed that five events are piled up.

providing a second copy. To cater for possible network interruptions, a disk buffer corresponding to the equivalent of 12 hours of heavy-ion data taking is foreseen in the DAQ cluster, and the equivalent of 2 days at the Tier 0. One copy of each reconstruction pass is stored on permanent storage at Tier 0 and external Tier 1s, most likely where they have been generated. One copy of Monte Carlo data is stored and distributed among the Tier 1s. The annual requirement for permanent storage in each Tier category is summarized in Table 7.8 on page 71. These values correspond to the needs during a standard year of running. Tier 2s are not required to provide permanent mass storage.

7.5.2 Transient storage

The requirements for transient storage on fast access media depend very much on the computing model, on the balance of available distributed processing resources, and on the network bandwidth and occupancy. In the absence of Grid simulation tools, our estimates on needed transient storage capacities are based on the requirement to store on disks data that will be frequently accessed primarily through non-scheduled tasks. They include:

- a fraction of the yearly raw data stored at Tier 0 (2%) and at each external Tier 1 (10%);
- one copy of the calibration and condition data at Tier 0 and each external Tier 1;
- two copies of the reconstructed data (ESD) of two reconstruction passes distributed onto Tier 0 and Tier 1s transient storage; a fraction of the ESD reside on disk at Tier 2.
- all analysis objects (AOD, event catalogue) distributed at Tier 2s and Tier 1s where they have been produced;
- one copy of generated Monte Carlo data are distributed in Tier 1s;
- two copies of reconstructed Monte Carlo distributed at Tier 2s where they have been produced.

The overall transient storage capacities required during a standard data taking year in each Tier category are listed in Table 7.8 on page 71.

7.6 Network

7.6.1 Tier 0

Data from the ALICE DAQ are transfered to Tier 0 at a rate of 1.25 GB/s during heavy-ion runs and 100 MB/s during pp runs. The DAQ farm provides for sufficient disk storage (50 TB) to buffer the equivalent of 12 hours of heavy-ion data taking. The maximum network bandwidth into Tier 0 is therefore

10 Gb/s during the heavy-ion run and 0.8 Gb/s during the pp run. The 10 Gb/s lines scheduled to be installed between the ALICE experiment and Tier 0 and exclusively dedicated to the raw data transfer will be sufficient.

Several kinds of transfer will contribute to the outgoing traffic from Tier 0 to external Tier 1s:

- the raw pp data export during the seven months of data taking leads to a rate of 55 MB/s;
- during the same time ESD data are exported at a rate of 3 MB/s;
- the raw heavy-ion data exportation during the month of data taking and the following four months of shutdown leads to a rate of 120 MB/s;
- during the four months of shutdown time ESD data are exported with a rate of 26 MB/s.

Data will be transferred simultaneously to every external Tier 1. This traffic results in a data transfer peak of 150 MB/s during the winter shutdown period. These values translate into a maximum outgoing network bandwidth of about 2 Gb/s (average over the year 1 Gb/s). A disk buffer of 200 TB is required at Tier 0 to store the equivalent of 48 hours of exported Pb–Pb RAW data.

7.6.2 Tier 1

The maximum incoming bandwidth required from Tier 0 into each Tier 1 is deduced from the discussion in the previous section to be 0.3 Gb/s assuming raw data are distributed in six Tier 1s. To the traffic from Tier 0 has to be added the traffic from Tier 2 exporting Monte Carlo data. Assuming that all Monte Carlo data are processed in Tier 2s and that there are on average 3.5 Tier 2s sending their locally produced data to a given Tier 1, this traffic amounts to 20 MB/s. The incoming bandwidth is therefore equal to about 2 Gb/s.

Tier 1 exports ESD data to Tier 2 for analysis. Assuming that all the analysis is done in Tier 2s and that each Tier 1 serves all the Tier 2s, the traffic from one Tier 1 to the Tier 2s amounts to 3 MB/s and requires a maximum bandwidth of about 0.03 Gb/s.

7.6.3 Tier 2

The incoming and outgoing bandwidth for Tier 2 is deduced from the discussion in the previous section to be 0.01 Gb/s and 0.6 Gb/s, respectively.

The requested network bandwidth for the various Tier categories is summarized in Table 7.8 on the next page. It should be noted that we assumed that the network is 100% efficient and operational 100% of the time. The network infrastructure should make provision for upgrades made possible by an evolving technology.

One element that makes this evaluation difficult is the duplication factor for disk files. It is clear that we cannot keep all the active data on disk without requiring an unreasonable amount of disk space. Data will be copied from Tier 1s permanent storage onto Tier 2s disk buffers. When the disk buffer is full, data will be removed and then copied again when needed. This may affect the network load between Tier 1s and Tier 2s and also between Tier 2s according to the end-user analysis pattern and the functionality of the middleware performing the file movement.

7.7 Ramp-up of resources

The total amount of resources required by ALICE for the production of Monte Carlo data and the processing of real data in a standard year of running, are summarized in Table 7.8 on the facing page. A small overall safety factor of 10% has been taken into account for all kinds of resources (CPU and storage). The staging of the offline resources (see Table 7.7) is dictated by the LHC heavy-ion schedule and is synchronized with the ramp-up of the DAQ online resources.

Table 7.7: Estimates for the computing resources ramp-up scenario. The timing of the resources increase is tuned to have the resources required in a given year available for the start of the heavy-ion run. The share of CERN CPU resources between Tier 0 and Tier 1/2 is describe in the text.

CPU (MSI2K)	2007	2008	2009	2010
CERN Tier 0	3.3	8.3	10.8	14.0
CERN Tier 1/2				
Ex Tier 1's	4.9	12.3	16.0	20.9
Ex Tier 2's	5.8	14.4	18.7	24.3
Total	14.0	35.0	45.5	59.2
Disk (TB)				
CERN Tier 0	95	238	309	402
CERN Tier 1/2	579	1447	1882	2446
Ex Tier 1's	2941	7353	9559	12426
Ex Tier 2's	2042	5106	6638	8629
Total	5658	14144	18387	23903
MSS (TB/year)				
CERN Tier 0	990	2475	3218	4183
CERN Tier 1	463	1158	1505	1957
Ex Tier 1's	2779	6947	9031	11740
Total	4232	10580	13754	17880

Before the first heavy-ion run, 20% of the nominal total resources are required to process pp data and to produce Monte Carlo data. ALICE foresees an early and short heavy-ion pilot run before the end of 2007 for which 40% of the resources required at CERN must be available already in mid 2007. 40% of the external resources must be available when the heavy-ion starts to take up the tasks of the CERN Tier 1 and Tier 2 which are not available during the heavy-ion run and during the first reconstruction pass (4 months after the heavy-ion run).

Table 7.8: Summary of the computing resources required by the ALICE computing model during a standard data-taking year (reference year 2008).

	CERN (Tier 0/1/2)	Tier 1s	Tier 2s	Total
CPU (MSI2k)	8.3	12.3	14.4	35.0
Transient storage (PB)	1.7	7.4	5.1	14.1
Permanent storage (PB/year)	3.6	7.0	–	10.6
Bandwidth in (Gb/s)	10	2	0.01	-
Bandwidth out (Gb/s)	1.2	0.03	0.6	-

The first full heavy-ion run is foreseen during the last quarter of 2008. Even though beams with reduced luminosity with respect to the nominal LHC luminosity might be available, data will be taken at the maximum rated allowed by the DAQ nominal bandwidth. In addition, even if the event size is smaller than the size used to estimate the resources required by the computing model, we will saturate the available DAQ bandwidth by tuning the triggers to increase the event rates, in such a way as to collect faster the statistics required for the ALICE physics goals. It would be unreasonable to limit the physics

reach of ALICE during the startup years by delaying the installation at CERN of the full computing capacity. The rampup of the external resources must follow the rampup of resources at CERN to avoid to accumulate data to be processed. Delaying the installation of external resources would deprive ALICE from the needed capacity to react quickly to unexpected discoveries. Therefore, 100% of the required resources must be installed at CERN, as well as 100% of the external resources, already in 2008 for the first Pb–Pb run.

In 2009 and 2010, a 30% increase is requested in Tier 1 and Tier 2 resources to be able to cope with the reconstruction of the data of the running year and previous years. The computing resources required at CERN include Tier 0, Tier 1 and Tier 2 type resources. The first pass heavy-ion reconstruction will be performed during the four months after the heavy-ion run at Tier 0 and requests all resources (7.5 MSI2k) available at the CERN Tier 0. During the same time no Tier 1/2 resources are requested at CERN.

7.8 Summary

The computing resources requirements in each Tier category are summarized in Table 7.8. Our resources deployment scenario from 2007 to 2010 is summarized in Table 7.7.

The resources at CERN are shared between Tier 0, Tier 1, and Tier 2 services. The resources at CERN Tier 1 are sized larger than the resources requested in an average Tier 1, but are similar to those foreseen in the large Tier 1s, such as GridKa. The resources at CERN Tier 2 are of the size of the resources requested in an average Tier 2. The resources at Tier 0 have been evaluated to process the pp data quasi-on line, as specified by our Computing Model. Just after the end of the heavy-ion run, Tier 1 and Tier 2 services will be switched off at CERN and their resources will be exclusively devoted to Tier 0. In such a way, sufficient resources will be made available to perform the first-pass reconstruction of heavy-ion data within the four months following the heavy-ion run. This mode of operation (see Fig. 7.1) will leave a sufficiently large time buffer allowing us to analyse data and to define the running conditions for the next heavy-ion run.

Tier 1s perform scheduled tasks such as additional reconstruction passes and scheduled analysis. Tier 2s produce and process Monte Carlo data and perform unscheduled end-user analysis tasks. The processing outside CERN is balanced to cope with the absence of Tier 1 and Tier 2 resources at CERN during the first-pass reconstruction of heavy-ion data and to provide an uniform global CPU resource request in external Tier 1s and Tier 2s. The yearly requirement for permanent storage is not likely to change with time, whereas the disk storage capacity requirements might change depending on the performance of the upcoming mass storage systems on the one hand and on the performance of the Grid on the other.

8 Computing Project organization and responsibilities

8.1 Computing Project responsibilities

The scope of the ALICE Computing Project is to provide the Collaboration with the framework, resources and software needed to extract the physics content from the data collected by the ALICE experiment and thus realize its physics potential and ultimate goal. The ALICE Computing Project is organized as shown in the chart of Fig. 8.1.

It has the ultimate responsibility for the coordination of the following activities, classified by responsibility:

1. Subdetector Software. This is the software for the simulation, reconstruction and analysis of the data coming from a subdetector and for its calibration and alignment. The responsibility for the development and maintenance of this software lies with the subdetector project.

2. Physics Analysis Software. This is the software that analyses the reconstructed data and extracts the physics results. The responsibility for this software rests with the Physics Working Groups organized by the Physics Board that is chaired by the Physics Coordinator.

3. Core computing

   (a) Core Software. This is the software that is common to all the subdetectors such as the transport Monte Carlo and the general framework for I/O, event processing, visualization, simulation, calibration, alignment and reconstruction. ALICE-specific services that interface with the Grid middleware belong to this category. Responsibility for this software rests with the Core Computing Project.

   (b) Infrastructure and Services. This includes the main areas:

      i. Central Support. Coordination, distribution and first line support of the ALICE software, coordination of data processing activities and the coordination of the ALICE Virtual Organization (VO) and distributed computing environment. It also covers relations with the LCG project.

      ii. Offline Coordination. Planning of the computing resources for the processing of ALICE data, management of relations with the computing centres and institutions providing these resources and with the LCG bodies that coordinate these resources.

   All these activities are a shared responsibility of the whole Collaboration, as they are services for all the ALICE physicists and are hosted by the Core Computing Project at CERN.

8.2 Computing Project organization

- The Computing Project (Fig. 8.2) is under the leadership of the Computing Coordinator, who is also the chair of the Computing Board..

- The Offline infrastructure is coordinated by the Offline Coordinator, who is also the chair of the Offline Board.

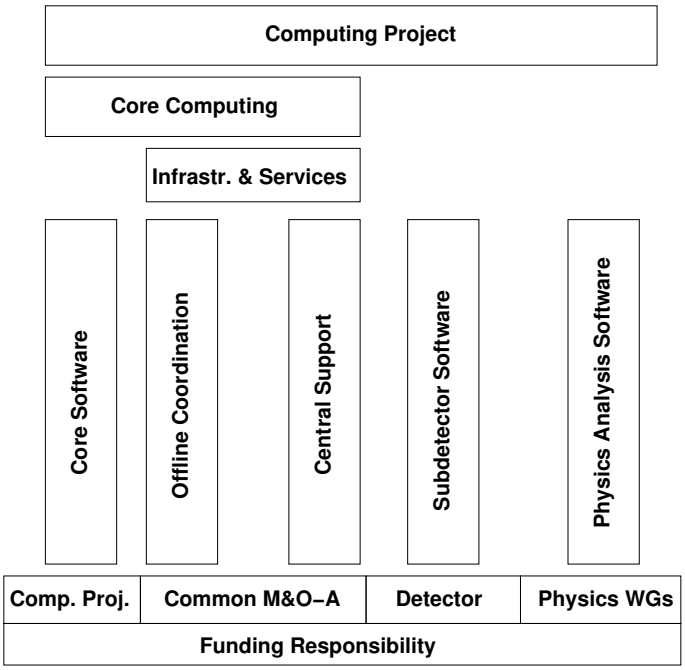

Figure 8.1: Organization of the ALICE Computing Project.

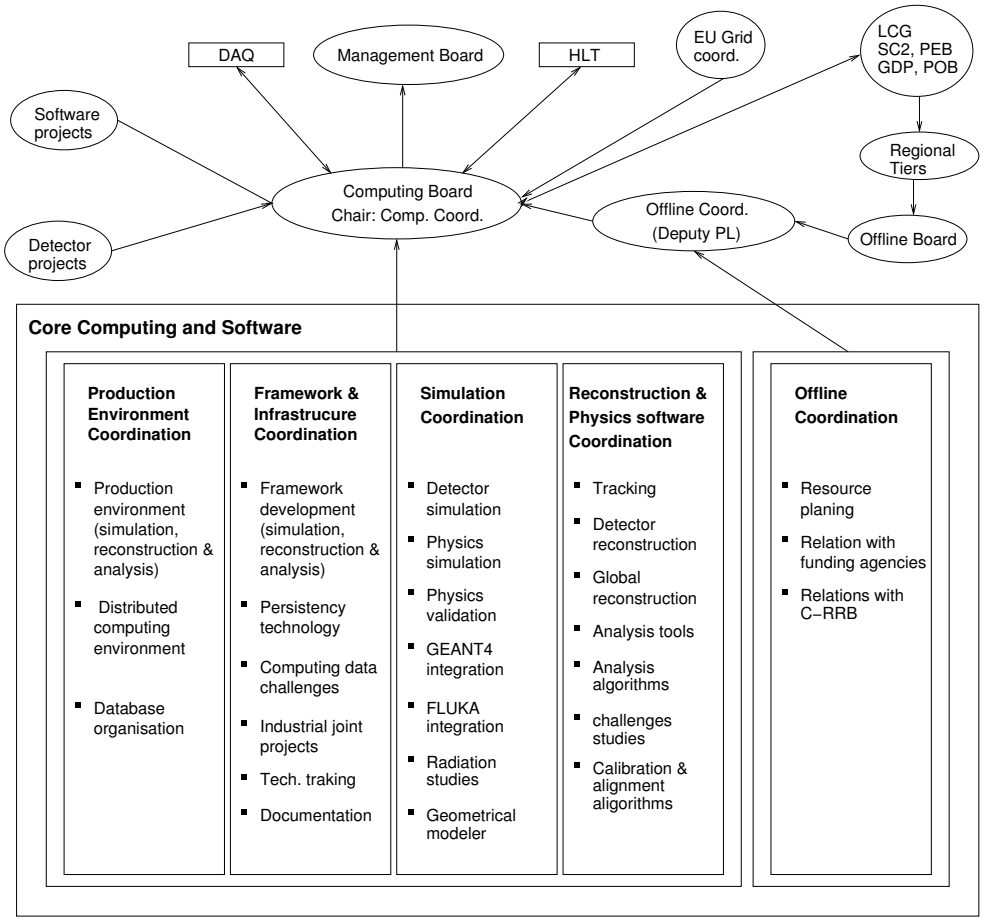

Figure 8.2: Organigramme of the Core Computing.

- Four Project Activity Area coordinators coordinate the software development: Simulation, Reconstruction and algorithms, Production Infrastructure and Databases, and Framework.
- Overall coordination for all project activities is realized through the Software and Computing Board (SCB) chaired by the Computing Coordinator and composed of the Offline Coordinator, the coordinators of the Computing Project Activity Areas, one or two representatives for each detector project, and one representative for major national computing facilities. The DAQ and HLT project leaders are ex officio members of the SCB.
- Coordination of the provision of the computing resources in collaborating institutions is performed via the Offline Board, chaired by the Offline Coordinator.
- Coordination with the ALICE Physics Working Groups is ensured through the ALICE Physics Coordinator.
- Representation within the ALICE Management is ensured through the Computing Coordinator being an ex officio member of the ALICE Management Board.
- The Computing and Offline coordinators ensure representation within the LCG project.

8.3 Organization of the Core Computing Project

The scope and responsibilities in the domain of Core Computing activity is described next.

1. Scope: The scope of the ALICE Core Computing Project (hereafter AC2) is defined as the development and maintenance of the experiment software framework and condition databases, the documentation and Web, the software infrastructure, visualization, and the production tools. It also includes ALICE software distribution and support, and its interfacing to the Grid and LCG software. Some areas of the project are shared with the ALICE DAQ and HLT project.
2. Responsibilities: The AC2 has the following responsibilities:
   (a) Design, prototyping, deployment, maintenance and documentation of the software framework.
   (b) Support for the possible central database services not provided by LCG.
   (c) First line support and distribution of all software produced. Detailed questions may need to be reported to the original authors or current maintainer of the software.
   (d) Development of a computing model and its validation in series of Physics and Computing Data Challenges. These are executed in collaboration with DAQ and HLT. These challenges imply organization and operation of large-scale testing of increasingly complex prototypes for the distributed production of simulated data, and the subsequent reconstruction and analysis of these data.
   (e) Implementation of the offline framework in the LCG Grid infrastructure and its interfacing with the LCG middleware.
   (f) Provision of continuously operational software enabling the physicists to assess the functionality of the framework toward the final goal of extracting physics from the data.
   (g) Liaison with the LCG project and with the regional centres providing the computing resources for ALICE.
   (h) Preparation of the ALICE Software and Computing TDR describing the ALICE Computing model.

(i) Review of the planned ALICE computing resource needs.

(j) Provide the Collaboration with the necessary justification of the present and planned computing needs to support negotiations with the funding agencies to obtain the computing resources.

(k) Preparation and update of the multiyear resource planning.

(l) Preparation and update of the multiyear manpower planning

(m) Review of the resources actually used by the Collaboration.

(n) Relations with the LCG management.

3. Project structure: ALICE has opted for a very lightweight AC2 team located mostly at CERN. The CERN team is constituted of a few personnel with long-term positions and a majority of personnel with short-term assignments. CERN and a few Collaboration institutes provide, on a voluntary basis, personnel for AC2, both locally and at CERN. CERN has taken the major responsibility for this activity, while well-identified sub-projects are executed either by institutes participating in the Computing Project having adequate skills (examples are the detector construction database project, the LCG integration, the Virtual Monte Carlo or the ALICE Web) or via collaboration with institutes not belonging to the Collaboration (e.g. for the coding rule checker project).

4. Project resources: The personnel for the Core Computing are people skilled in physics data processing and simulation, but also in areas such as OO analysis and design, C++ and other languages, databases and data management systems, computing systems, software process, quality control etc. In ALICE a large majority of these people are trained physicist.

8.4 Activities in the Core Computing Project

The complete activities in the ALICE Core Computing are described in task-oriented Activity Areas.

- **AA1**: Project (Computing & offline) coordination
- **AA2**: Framework development
- **AA3**: Simulation coordination
- **AA4**: Reconstruction coordination
- **AA5**: Analysis tools coordination
- **AA6**: Databases and production infrastructure
- **AA7**: Production and quality assurance
- **AA8**: Program librarian
- **AA9**: Persistency and computing data challenge
- **AA10**: System support
- **AA11**: Radiation studies
- **AA12**: Documentation and Web
- **AA13**: Detector construction database
- **AA14**: LCG integration

8.5 Institutes participating in the activities

The current institutes' participation in the Core Offline activities are listed in Table 8.1 together with their major contribution.

Table 8.1: List of institutes participating in the Core Offline activities and their major contribution.

Institute	Activity area	Main responsibility
CERN	AA1,2,3,4,5,6,7,8,9,10,12	Overall project coordination
		Core software as defined above
CEADEN Cuba	AA2,12	Interface with detector simulation transport programs;
		Maintenance of the Web site
INFN Torino	AA14	Integration of ALICE software with LCG Grid
IN2P3	AA1,3	Overall project coordination,
		GEANT 4 integration
Kosice, Slovakia	AA11	Radiation studies
Warsaw TU	AA6,13	Detector construction DB
Sejong, Korea	AA7	Grid middleware and production support

8.6 Milestones

The major milestones (MS) of the Computing Project are those linked with the execution of the two remaining Physics Data Challenges (PDCs) until the start of data taking in the first semester of 2007.

- **MS1–May 2005**: PDC05 – Start of event production (phase 1)

  PDC05 will use all resources available on the Grid and access them through the LCG middleware and the AliEn services.

- **MS2–June 2005**: AliRoot framework release.

  The framework will include the following items:

  - a prototype for the Condition infrastructure;
  - the FLUKA interface to the Virtual Monte Carlo and FLUKA fully validated to be used as the main tracking package;
  - the ROOT Geometrical Modeller as unique package for the detector geometry description.

- **MS3–June 2005**: Computing TDR (the present document) submitted to the LHCC.

- **MS4–July 2005**: PDC05 – Start of combined test with SC3 (phase 2).

  Phase 2 of PDC05 (see Chapter 3) is started together with the throughput test of the LCG Service Challenge 3 (SC3). In case SC3 is delayed, the PDC05 phase 2 schedule will not be modified.

- **MS5–September 2005**: PDC05 – Start of distributed analysis (phase 3).

  Phase 3 of PDC05 (see Chapter 3) is combined with the services test of LCG SC3. It includes the AliEn services and gShell, the ALICE user interface to the AliEn services.

- **MS6–September 2005**: Metadata prototype ready.

- **MS7–December 2005**: Condition infrastructure deployed.

- **MS8–December 2005**: Preliminary implementation of algorithms for alignment and calibration ready for all detectors.
- **MS9–January 2006**: Release of AliRoot framework in preparation of the PDC06. Final prototype of alignment and calibration ready to be tested. Alignment and calibration algorithms prepared for all detectors. Global alignment and inter-calibration of the detectors at the prototype stage.
- **MS10–January 2006**: Start of PDC06. Test of the full production chain, including calibration and alignment. Size approximately equal to 20% of the real data of a standard data-taking year. Distributed batch analysis of the data. Evalation of the distributed interactive analysis framework.
- **MS11–June 2006**: End of PDC06. Evaluation of the data challenge results. Planning of most urgent activities for next year.
- **MS12–June 2006**: Final implementation of algorithms for alignment and calibration ready for all detectors. This includes inter-calibration and alignment algorithms.
- **MS13–December 2006**: ALICE computing environment ready for data taking.

References

Summary

[1] http://www.cern.ch.
[2] http://lhc-new-homepage.web.cern.ch/lhc-new-homepage.
[3] LHCC Review of Computing Resources for the LHC Experiments,
CERN-C-RRB-2005-006;
CERN-LHCC-2005-006;
LHCC-G-089.
[4] CERN/LHCC 2003-049, ALICE Physics Performance Report, Volume 1 (7 November 2003);
ALICE Collaboration: F. Carminati *et al.*, J. Phys. G: Nucl. Part. Phys. **30** (2004) 1517–1763.
[5] http://monarc.web.cern.ch/MONARC.
[6] http://lcg.web.cern.ch/LCG/.
[7] http://alien.cern.ch.

Chapter 1

[1] ALICE Collaboration, *Technical Design Report of Trigger, Data Acquisition, High-Level Trigger and Control System*, CERN/LHCC/2003–062.
[2] The ALICE Trigger Project, ALICE Central Trigger Processor, User Requirement Document, Draft 1.0, 24 October 2001.
[3] The ALICE Trigger Project, ALICE Local Trigger Unit, Preliminary Design Review Document, Revision 0.1, 1 September 2002.
[4] B. G. Taylor, LHC machine timing distribution for the experiments, in *Proc. of the 6th Workshop on Electronics for LHC Experiments*, Cracow, Poland, 2000 (CERN 2000-010, Geneva, 2003).
[5] J. Baud *et. al.*, CASTOR status and evolution, in *Proc. Conf. on Computing in High-Energy Physics*, La Jolla, CA, USA, March 2003 (SLAC, Stanford).
[6] ALICE DAQ, DATE V4 User's Guide, Internal Note ALICE-2002-036.
[7] R. Divià *et al.*, Data Format over the ALICE DDL, Internal Note ALICE-INT-2002-10 V4.
[8] ALICE DAQ, ALICE DDL Interface Control Document, Internal Note ALICE-INT-1996-43 V9.1.
[9] ALICE DAQ, ALICE DDL Hardware Guide for the Front-end Designers, Internal Note ALICE-INT-1998-21 V1.3.
[10] J. Christiansen *et al.*, TTCrx Reference Manual A Timing, Trigger and Control Receiver ASIC for LHC detectors, V 3.8, RD12 project, January 2003.
[11] B. G. Taylor, TTC Machine Interface (TTCmi) User Manual, Rev 1.3, RD12 project.
[12] P. Farthouat and P. Gällnö, TTC-VMEbus Interface TTCvi MkII, Rev 1.1, RD12 project, May 2000.
[13] R. Brun *et. al.*, *Computing in ALICE*, Nucl. Instr. Meth. **A502** (2003) 339-346.
[14] W. Carena *et al.*, Online performance monitoring of the Third ALICE Data Challenge (ADC III), in *Proc. Conf. on Nuclear Electronics & Computing*, Varna, Bulgaria, September 2001.
[15] T. Anticic *et al.*, Challenging the challenge: handling data in the Gigabit/s range, to be published in *Proc. Conf. on Computing in High-Energy Physics*, La Jolla, CA, USA, March 2003 (SLAC, Stanford).

Chapter 2

[1] http://www.cern.ch/ALICE/Projects/offline/aliroot/Welcome.html.

[2] See for instance G. Booch, *Object-oriented Analysis and Design with Applications, 2nd edition.*, (Benjamin Cummings, Redwood City, 1993) ISBN 0-8053-5340-2.

[3] http://wwwinfo.cern.ch/asd/cernlib.

[4] http://wwwinfo.cern.ch/asd/paw.

[5] R. Brun, F. Bruyant, M. Maire, A.C. McPherson, P. Zanarini, GEANT3 User Guide, CERN Data Handling Division DD/EE/84–1 (1985);
http://wwwinfo.cern.ch/asdoc/geantold/GEANTMAIN.html.

[6] http://root.cern.ch.

[7] http://lcg.web.cern.ch/LCG.

[8] M. Ballintijn, R. Brun, F. Rademakers and G. Roland, Distributed parallel analysis framework with PROOF, Proc. of TUCT004.

[9] P. Saiz *et al.*, Nucl. Instrum. Methods **A502** (2003) 437–440;
http://alien.cern.ch/.

[10] A. Fassò *et al.*, in *Proc. of Computing in High Energy and Nuclear Physics*, La Jolla, California (2003); http://www.slac.stanford.edu/econf/C0303241/proc/papers/MOMT004.PDF.

[11] S. Agostinelli *et al.*, Geant4 - A simulation toolkit, CERN-IT-20020003, KEK Preprint 2002-85, SLAC-PUB-9350, submitted to Nucl. Instrum. and Methods A.
http://wwwinfo.cern.ch/asd/geant4/geant4.html.

[12] I. Hřivnáčová *et al.*, in *Proc. of Computing in High Energy and Nuclear Physics*, La Jolla, California (2003); http://www.slac.stanford.edu/econf/C0303241/proc/papers/THJT006.PDF.

[13] R. Brun, A. Gheata and M. Gheata, The ROOT geometry package, NIM **A502** (2003) 676–680.

[14] http://www.star.bnl.gov.

[15] A. Shoshani, A. Sim, and J. Gu, Storage resource managers: Middleware components for grid storage, in *Proceedings of the Nineteenth IEEE Symposium on Mass Storage Systems and Technologies*, IEEE, New York (2002).

[16] K. Wu, W.-M. Zhang, A. Sim, J. Gu and A. Shoshani, Grid collector: An event catalog with automated file management in *Proceedings of IEEE-NSS*, IEEE, Portland, (2003).

[17] LHC Computing Review, CERN/LHCC/2001–004.

[18] See for instance the results of the LCG Baseline Services study group
http://lcg.web.cern.ch/LCG/peb/BS/.

[19] See for instance:
K. Beck, C. Andres, *Extreme programming explained: embrace change*, (Addison-Wesley Professional; 2nd edition (November 16, 2004)), ISBN-0-3212-7865-8.

[20] http://alisoft.cern.ch/offline/aliroot-pro/nightbuilds.html.

[21] http://www.cvshome.org.

[22] http://alisoft.cern.ch/cgi-bin/cvsweb.

[23] http://alisoft.cern.ch/offline/aliroot-new/roothtml/USER_Index.html.

[24] W. J. Brown *et al.*, *AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis* (Wiley; 1st edition (March 20, 1998)), ISBN-0-4711-9713-0.

[25] http://v.mahon.free.fr/pro/freeware/memcheck.

[26] http://valgrind.org.

[27] http://www.intel.com/software/products/vtune.

[28] IRST – Trento, Italy is a research centre conducting research in computer science, micro-systems and surface physics. The code checker is now a production tool used by ALICE, ATLAS, IT-Control (PVSS), Root, and others.
http://www.itc.it/irst/Renderer.aspx?targetID=111.

[29] P. Tonella and Alessandra Potrich, Reverse engineering of the interaction diagrams from C++

code, in *Proceedings of ICSM 2003, International Conference on Software Maintenance*, pp. 159-168, Amsterdam, The Netherlands, September 2003;
P. Tonella and A. Potrich, Static and dynamic C++ code analysis for the recovery of the object Ddagram, in *Proceedings of ICSM 2002, International Conference on Software Maintenance*, pp. 54-63, Montreal, Canada, October 2002;
P. Tonella and A. Potrich, Cjj: a subset of C++ compliant with Java, Science of Computer Programming, vol. 42/2-3, pp. 229-271, January 2002;
P. Tonella and Alessandra Potrich, Reverse engineering of the UML class diagram from C++ code in presence of weakly typed containers, in *Proceedings of ICSM 2001, International Conference on Software Maintenance*, pp. 376-385, Florence, Italy, November 7-9, 2001;
A. Potrich and P. Tonella, C++ code analysis: an open architecture for the verification of coding rules, in . *Proceedings of CHEP'2000, Int. Conf. on Computing in High Energy and Nuclear Physics*, pp. 758-761, Padova, Italy, February 7-11, 2000.

[30] P. Tonella, A. Potrich, *Reverse Engineering of Object Oriented Code*, (Springer; 1st edition (December 17, 2004)), ISBN: 0-3874-0295-0.

[31] http://alisoft.cern.ch/offline/codingconv.html.

[32] http://alisoft.cern.ch/offline/aliroot-new/log/violatedRules.html.

[33] http://www.rational.com/uml.

[34] http://alisoft.cern.ch/offline/reveng.

Chapter 3

[1] P. Saiz *et al.*, Nucl. Instrum. Methods **A502** (2003) 437-440;
http://alien.cern.ch/.

[2] http://lcg.web.cern.ch/LCG/.

[3] http://lhc-new-homepage.web.cern.ch/lhc-new-homepage/.

[4] http://www.cern.ch/MONARC/.

[5] The Grid: Blueprint for a New Computing Infrastructure I. Foster, C. Kesselmann (Eds), M. Kaufmann, 1999;
I. Foster, C. Kesselman, S. Tuecke, The Anatomy of the Grid Enabling Scalable Virtual Organizations, see, http://www.globus.org/research/papers/anatomy.pdf.

[6] http://egee-intranet.web.cern.ch/egee-intranet/gateway.html.

[7] http://www.w3.org/2002/ws/.

[8] http://lcg.web.cern.ch/LCG/peb/arda/Default.htm.

[9] http://root.cern.ch/.

[10] M. Ballintijn, R. Brun, F. Rademakers and G. Roland, *Distributed Parallel Analysis Framework with PROOF*, Proc. of TUCT004.

Chapter 4

[1] CERN/LHCC 2003-049, ALICE Physics Performance Report, Volume 1 (7 November 2003);
ALICE Collaboration: F. Carminati *et al.*, J. Phys. G: Nucl. Part. Phys. **30** (2004) 1517–1763.

[2] X. N. Wang and M. Gyulassy, Phys. Rev. **D44** (1991) 3501;
M. Gyulassy and X. N. Wang, Comput. Phys. Commun. **83** (1994) 307–331;
the code can be found in http://www-nsdth.lbl.gov/∼xnwang/hijing/.

[3] J. Ranft, Phys. Rev. **D51**, (1995) 64;
J. Ranft, New features in DPMJET version II.5,hep-ph/9911213;
J. Ranft, DPMJET version II.5, code manual, hep-ph/9911232.

[4] H.-U. Bengtsson and T. Sjostrand, Comput. Phys. Commun. **46** (1987) 43;
the code can be found at http://nimis.thep.lu.se/∼torbjorn/Pythia.html .
T. Sjostrand, Comput. Phys. Commun. **82** (1994) 74;
the code can be found at http://www.thep.lu.se/∼torbjorn/Pythia.html .
[5] F. Abe *et al.*, (CDF Collaboration), Phys. Rev. Lett. **61** (1988) 1819.
[6] R. Brun, F. Bruyant, M. Maire, A.C. McPherson, P. Zanarini, GEANT3 User Guide, CERN Data Handling Division DD/EE/84–1 (1985);
http://wwwinfo.cern.ch/asdoc/geantold/GEANTMAIN.html .
[7] S. Agostinelli *et al.*, Geant4 - A simulation toolkit, CERN-IT-20020003, KEK Preprint 2002-85, SLAC-PUB-9350, submitted to Nucl. Instrum. and Methods A;
http://wwwinfo.cern.ch/asd/geant4/geant4.html .
[8] A. Fassò *et al.*, in *Proc. of Computing in High Energy and Nuclear Physics*, La Jolla, California (2003); http://www.slac.stanford.edu/econf/C0303241/proc/papers/MOMT004.PDF .
[9] I. Hřivnáčová *et al.*, in *Proc. of Computing in High Energy and Nuclear Physics*, La Jolla, California (2003); http://www.slac.stanford.edu/econf/C0303241/proc/papers/THJT006.PDF .
[10] R. Brun, A. Gheata, and M. Gheata, *Proc. of Computing in High Energy and Nuclear Physics*, La Jolla, California (2003);
http://www.slac.stanford.edu/econf/C0303241/proc/papers/THMT001.PDF .
[11] See for instance the note ATL-PHYS-96-086
(http://preprints.cern.ch/cgi-bin/setlink?base=atlnot&categ=Note&id=phys-96-086).
[12] I. González Caballero *et al.*, *Proc. of Computing in High Energy and Nuclear Physics*, La Jolla, California (2003);
http://www.slac.stanford.edu/econf/C0303241/proc/papers/MOMT011.PDF .
[13] F. Carminati, I. González Caballero, Geant4: A benchmark of hadronic processes, ALICE-INT-2001-041.
[14] Note in course of publication, see
http://wwwinfo.cern.ch/asd/geant4/reviews/delta-2002/schedule.html .
[15] B. Pastircak and A. Morsch, ALICE-INT-2002-028.
[16] A. Morsch, ALIFE: A geometry editor and parser for fLUKA, ALICE-INT-1998-029.
[17] F. Carminati and A. Morsch, *Proc. of Computing in High Energy and Nuclear Physics*, La Jolla, California (2003);
http://www.slac.stanford.edu/econf/C0303241/proc/papers/TUMT004.PDF .
[18] http://aldwww.cern.ch .
[19] http://nuclear.ucdavis.edu/ jklay/ALICE/ .
[20] G. Tsiledakis *et al.*, ALICE-INT-2003-010.
[21] A. Fassò, A. Ferrari, P.R. Sala, G. Tsiledakis, Implementation of xenon capture gammas in FLUKA for TRD background calculation, ALICE-INT-2001-28.
[22] A. Dainese and R. Turrisi, ALICE-INT-2003-019;
A. Dainese and R. Turrisi, ALICE-INT-2003-028.

Chapter 5

[1] ALICE Physics Performance Report Volume 2 (to be published).
[2] P. Billoir, Nucl. Instrum. Meth. **A225** (1984) 352;
P. Billoir *et al.*, Nucl. Instrum. Meth. **A241** (1985) 115;
R. Fruhwirth, Nucl. Instrum. Meth. **A262** (1987) 444;
P. Billoir and S. Qian, Nucl. Instrum. Meth. **A294** (1990) 219.
[3] F. Riggi *et al.*, ALICE-INT-2001-26.
[4] ALICE Collaboration, *Technical Design Report of the Time-Projection Chamber*, CERN/LHCC/2000-01.

[5] A. Zintchenko *et al.*, ALICE-INT-2003-006 v.1.
[6] L. A. Shepp and Y. Vardi, IEEE Trans. Med. Im., v.1 (1982) 113.
[7] M. Aguilar-Benitez *et al.*, Z. Phys. **C50** (1991) 405–426.
[8] A. Andronic *et al.*, Nucl. Instrum. Meth. **A522** (2004) 40.
[9] ALICE Collaboration, *Technical Design Report of the Photon Spectrometer (PHOS)*, CERN/LHCC/1999-04.
[10] ALICE Collaboration, *Technical Design Report of Trigger, Data Acquisition, High-Level Trigger and Control System*, CERN/LHCC/2003–062.
[11] A. Vestbø, PhD Thesis, University of Bergen (2004).
[12] C. Cheshkov, ALICE Internal Note (to be published).

Chapter 6

[1] CERN/LHCC 2003-049, ALICE Physics Performance Report, Volume 1 (7 November 2003); ALICE Collaboration: F. Carminati *et al.*, J. Phys. G: Nucl. Part. Phys. **30** (2004) 1517–1763.
[2] http://alien.cern.ch/download/current/gClient/gShell_Documentation.html.
[3] http://glite.web.cern.ch/glite/.
[4] http://root.cern.ch/root/PROOF.html.
[5] http://root.cern.ch.
[6] CERN/LHCC 99-12.
[7] CERN/LHCC 2000-001.
[8] A. Dainese, PhD Thesis, University of Padova, 2003, [arXiv:nucl-ex/0311004].
[9] A. Stavinsky *et al*, NUKLEONIKA **49** (Supplement 2) (2004) 23–25; http://www.ichtj.waw.pl/ichtj/nukleon/back/full/vol49_2004/v49s2p023f.pdf.
[10] P.K. Skowroński for ALICE Collaboration, [arXiv:physics/0306111].
[11] R. Lednický and V.L. Lyuboshitz, Sov. J. Nucl. Phys. **35** (1982) 770.
[12] http://www.nscl.msu.edu/ pratt/freecodes/crab/home.html.
[13] C. Loizides, PhD Thesis, University of Frankfurt, 2005, [arXiv:nucl-ex/0501017].

Chapter 7

[1] LHC Computing Review, CERN/LHCC/2001-004.
[2] LHCC Review of Computing Resources for the LHC Experiments, CERN-C-RRB-2005-006; CERN-LHCC-2005-006; LHCC-G-089.
[3] ALICE Collaboration, *Technical Design Report of Trigger, Data Acquisition, High-Level Trigger and Control System*, CERN/LHCC/2003–062.
[4] CERN/LHCC 2003-049, ALICE Physics Performance Report, Volume 1 (7 November 2003); ALICE Collaboration: F. Carminati *et al.*, J. Phys. G: Nucl. Part. Phys. **30** (2004) 1517–1763.

Chapter 8

[1] LHC Computing Review, CERN/LHCC/2001-004.