The EU Simulation Environment

Ties Behnke, SLAC/DESY

- **BRAHMS**: our trusted, old full simulation package
  - FORTRAN
  - GEANT321
  - grown, not designed

- The new world:
  - MOKKA
  - GEANT4
  - Object oriented

- **SIMDET**: the fast simulator
  - FORTRAN
  - fast
  - Fairly complete
  - one detector: TESLA TDR

- **SGV**: fast Simdet alternative
We are working on defining a common hit storage format between BRAHMS and MOKKA, and between the European and the US frameworks.
The BRAHMS suite contains two programs:

- GEANT3 based simulation code ("BRAHMS proper")
- The reconstruction program REREKO

**BRAHMS**: The *simulation* program:

- complete implementation of the TDR tracker
- full implementation of the TDR calorimeter
- full implementation of the forward system
- full implementation of the muon system

**REREKO**: The *reconstruction* program

- full track reconstruction and detector merging code
- calorimeter reconstruction code
- full energy flow algorithms (a version of..)

most code is still FORTRAN based

technical detail:
simulation and reco may be run together or separately

communication between simulation and reconstruction via simple serial gzipped, files
Some details

Tools for event display and online event investigation are available.
Some Physics Example

Reconstruction of top quark in fully hadronic decay channel:

- Full simulation of signal
- Background using smear mode MC
- No cheating

BRW $\oplus$ Gauss

Peak = $175.37 \pm 0.76$ GeV
BRW width = $7.7$ GeV
Gauss width = $9.5 \pm 0.4$ GeV
candidates = 855

$\sqrt{s} = 800$ GeV

S. Chekanov, V. Morgunov
MOKKA

MOKKA is a GEANT4 based full simulation framework (main author: P. Moras de Freitas)

MOKKA has
- A database based geometry system
- Detailed calorimeter simulation
- Tracking detector simulation similar to BRAHMS
- No reconstruction

Vertex Detector (VXD)  Si Intermediate Tracker (SIT)

Forward Tracking Disks (FTD)  Beam pipe (Tube)
Reconstruction behind MOKKA

- MOKKA writes out hits in a simple ASCII format
- Hits are read by reconstruction code
- Currently existing packages (as far as I know)
  - Energy flow reconstruction algorithm (REPLIC, author J.C.Brient)
  - Topological reconstruction in the calorimeter (author P. Gay)
- At the moment there is no full tracking reconstruction package available.
**SIMDET** is a parametric Monte Carlo program to simulate the detector at TESLA

Main detector components are implemented according to the TESLA technical design report (TDR), with

- a Vertex Detector
- a tracking System
- electromagnetic and hadronic calorimeters, low angle calorimeter and tagger

Using results from the ab initio Monte Carlo program BRAHMS, track parameters and calorimetric deposits are treated in a realistic way. Pattern recognition is emulated using cross references between generated particles and detector response. An energy flow algorithm defines the output of the program.
SGV: fast/slow Monte Carlo

- SGV (author M. Beggren): fast/slow simulation
  - Analytic tracking of particles through detector and fields
  - Simulation of hits along the track
  - Simulation of response of calorimeter
  - Allows the use of realistic tracking packages (not currently implemented)

- Plans for SGV:
  - Currently FORTRAN based system
  - Transition to OO is starting
  - Michael wants to translate SGV into C/ C++

- SGV is available from the ECFA/DESY simulation WEB pages
News/ Changes II

- dE/dx implemented
- IP constraint (optional) implemented
- new CIRCE version
- PYTHIA 6.1 interface implemented
- CLIC version available
- updated note / user guide available

SIMDET Version 4.0 is available from the WEB
(CVS depository zt Zeuthen)
SIMDET acknowledgments

Thorsten Ohl  
beamstrahlungs code CIRCE

Pable Garcia  
PHYTHIA 6.125 implementation

Chris Damerell  
CCD vertex detector resolutions

Marco Battaglia  
APS vertex detector resolutions

CLIC version

Klaus Mönig  
BRAHMS tracker resolution,
parametrisation of the covariance matrix

Michael Hauschild  
dE/dx code

Vassili Djordjadze  
BRAHMS ECAL/HCAL response;
dE/dx tracking routines

Norbert Tesch,  
Karsten Büßer  
LCAL/ LAT resolutions

Harald Vogt  
WWW implementation and many discussions

M. Pohl and H.J. Schreiber
Next Steps

- basic program development done
- user interface now there (needs to be finalised / extended/ discussed)

- need detailed and systematic performance studies
  
  ongoing: physics studies (top mass resolution, W mass resolution etc)
  
  ➡️ results (prel) available

- need much more low-level studies:
  - photon ID
  - hadron ID
  - cluster resolution
  - dependence on internal parameters
  - etc etc.

- Need comparison to GEANT4 studies (MOKKA)
The Goal

- The goal: develop and maintain a modern simulation environment which is
  - flexible
  - maintainable for a long time to come
  - scalable

- At the same time:
  - continue the support for the existing system for still some time to come

- Ideally: maintain a link between the programs to avoid duplication and translation errors

Fortran → object orientation

BRAHMS 3xx

GEANT3 kernel
• tracking package
• calorimeter package

GEANT4 kernel
• object oriented tracking package
• object oriented calorimeter package

common hit format: LCIO
Technical Issues: Formats

- Hits after the simulation and digitisation:
  - stored in “HITS” file

  BRAHMS: binary output (c-write routines)
  gzip on the fly (ZLIB algorithm)

basic file structure:
- Start of Run record: run/ event number
- “constant record”: detector configuration at simulation time

Start of Event Marker

Subdetector Hits: Identifier, number of words/event
- hit record (typically x,y,z,E, but can be more)
- MC particle record (MC only)
- GEANT Volume record (MC only)

End of Event Marker

Record is at some level self-describing (adding / removing detectors..)
User has to interpret however the information ....
The HITS format

There is a clear need for a standardized format to exchange data between different simulation packages

A discussion on such a format has started.

An agreement has been reached with our american collegues

Work on implementing this agreement is ongoing

Proposal:
- simple packed format, portable, implemented for FORTRAN, C++, JAVA
- mechanism for simple pointer support
- mechanism for “direct access”

We are looking at an extension of the american SIO format (very similar to the current BRAHMS format)

LCIO (see Tony's talk)
The Final Output

- After reconstruction: store “energy flow objects” as main items:
  - BRAHMS: adopt scheme already implemented in SIMDET, with some (minor) modifications

### MC record

<table>
<thead>
<tr>
<th>Status</th>
<th>Parameters</th>
<th>Pointers</th>
<th>Tracks</th>
<th>Calorimeter</th>
<th>Muon</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>px, momentum</td>
<td>MC record num!Track Parameters ECAL Muon track/E info</td>
<td>Energy Fraction, theta, phi, qHCA! error matrix</td>
<td>repeated ngen</td>
<td>repeated ntrk</td>
</tr>
<tr>
<td>MC ID</td>
<td>py</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NGEN</td>
<td>pz</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NTRK</td>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NCAL</td>
<td>m</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NMUON</td>
<td>Q</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Eflow record

- Pointers to calorimeter hits
- Track parameters: p, theta, phi, q, d0, z0, energy fraction, dE/dx information

Energy Flow Workshop, Nov 7-9, 2002  Ties Behnke: Simulation in Europe  16
Combining MOKKA and BRAHMS

- First version of link between MOKKA (Geant4) simulation and BRAHMS reconstruction does exist:
  - Tracker working (though there are problems with the MOKKA geometry)
  - ECAL working
  - HCAL still problematic (tile geometry not implemented in MOKKA)

- Further developments:
  - LCIO package (T. Johnson's talk)

- Such links can save a lot of work in re-implementing packages in different environments

Plot: Frank Gaede, DESY Hamburg
Accessing the information

- The output information is written to file: same format as hit file
  - can read with stand-alone application (FORTRAN, JAVA, C ...)
  - a JAVA interface is under development

- Internally the information is available through FORTRAN statement functions
  - simple array-like syntax, maps onto internal ZEBRA store
  - hides ZEBRA completely from the user
  - allows the use of “pointers” in FORTRAN
  - fast

```fortran
  do neflow=1, nef(1)
    px = ref(1,nef)
    nehit = ief_ne(neflow)
    if ( ref(8,nef) ne 0. ) then
      ............
    end if
  end do
```

Internally the link to the hit information is maintained:
- track hit info
- calo hit info
- muon hit info ...
For non-FORTRAN users

Discussions have started to define a data model to be used in C++ and Java

ReconstructedParticle

With pointers to MC info and more detailed tracking and calo info

Goal: provide a unified interface (as similar as possible) for JAVA (JAS) and C++ (ROOT) users: LCIO package

Try to coordinate this between MOKKA and LCJAS and LCROOT developers
Summary

The “old fashioned” simulation programs BRAHMS and SIMDET are available and operational. They are ready for analysis work.

New tools are under development. MOKKA is becoming more complete.

Reconstruction in the MOKKA frame are starting to be available.

We seek close collaboration with all developers for a more integrated simulation environment.

All tools are available from our Simulation WEB page:

http://www-zeuthen.desy.de/linearCollider/