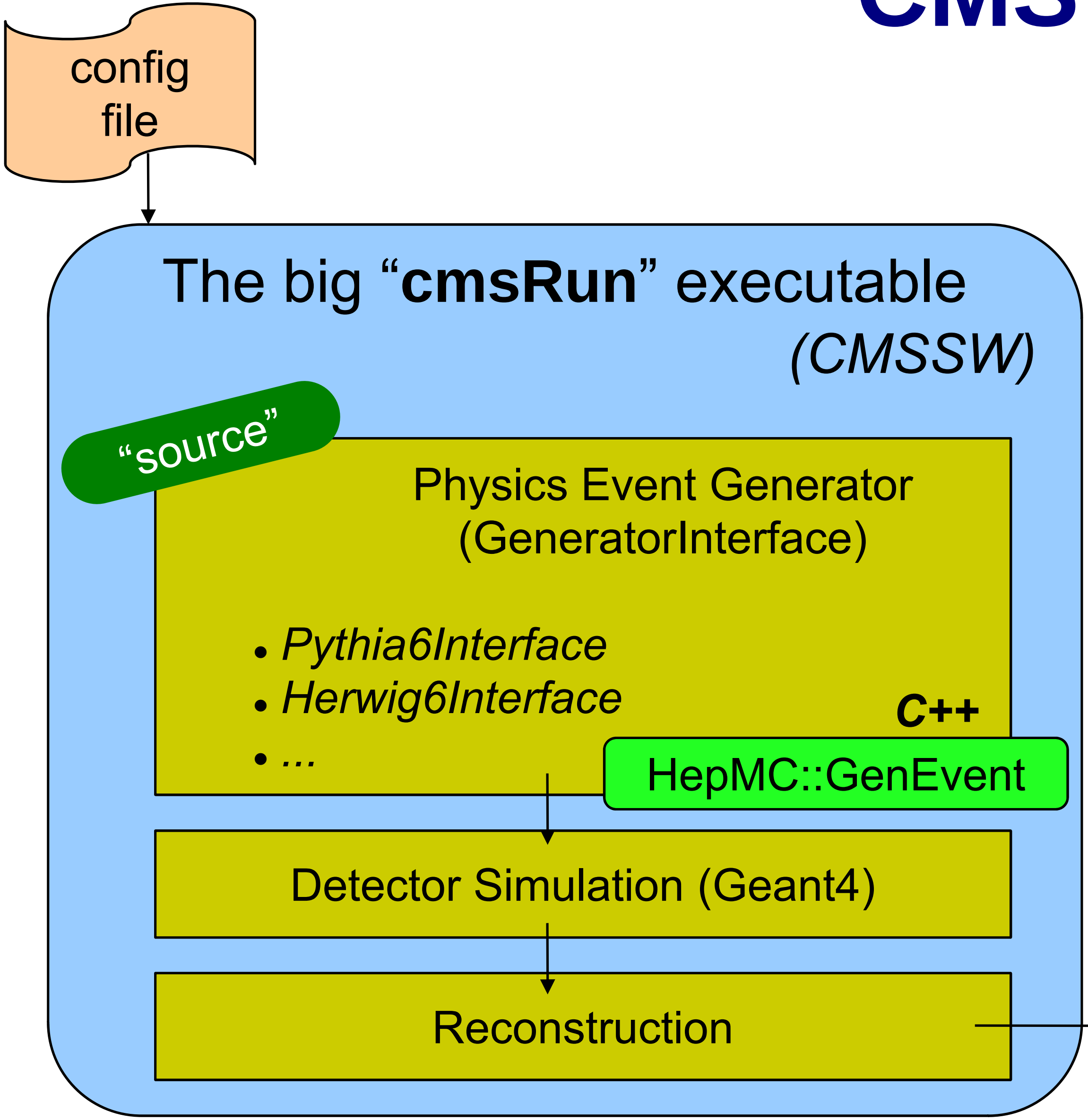
handling of LHE files in the CMS production and usage of MCDB

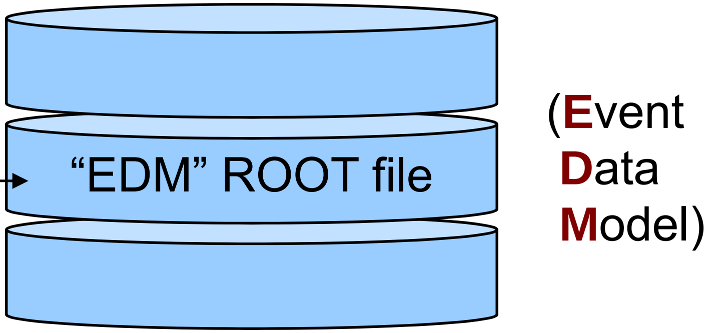
Christophe Saout
CERN, University of Karlsruhe

on behalf of the CMS physics event generators group

Traditional MC Production in CMS


config file
The big "**cmsRun**" executable *(CMSSW)*
"source"
Physics Event Generator (GeneratorInterface)
• *Pythia6Interface*
• *Herwig6Interface*
• ...
C++
HepMC::GenEvent
Detector Simulation (Geant4)
Reconstruction

- One big **CMS executable** for everything
- Steered by **config files**
- One generic data file format for everything: "**EDM**" *(based on ROOT)*
- Built out of **modules** *(shared libraries)*
- Source can either be:
  - an **Event Generator**
  - another EDM file

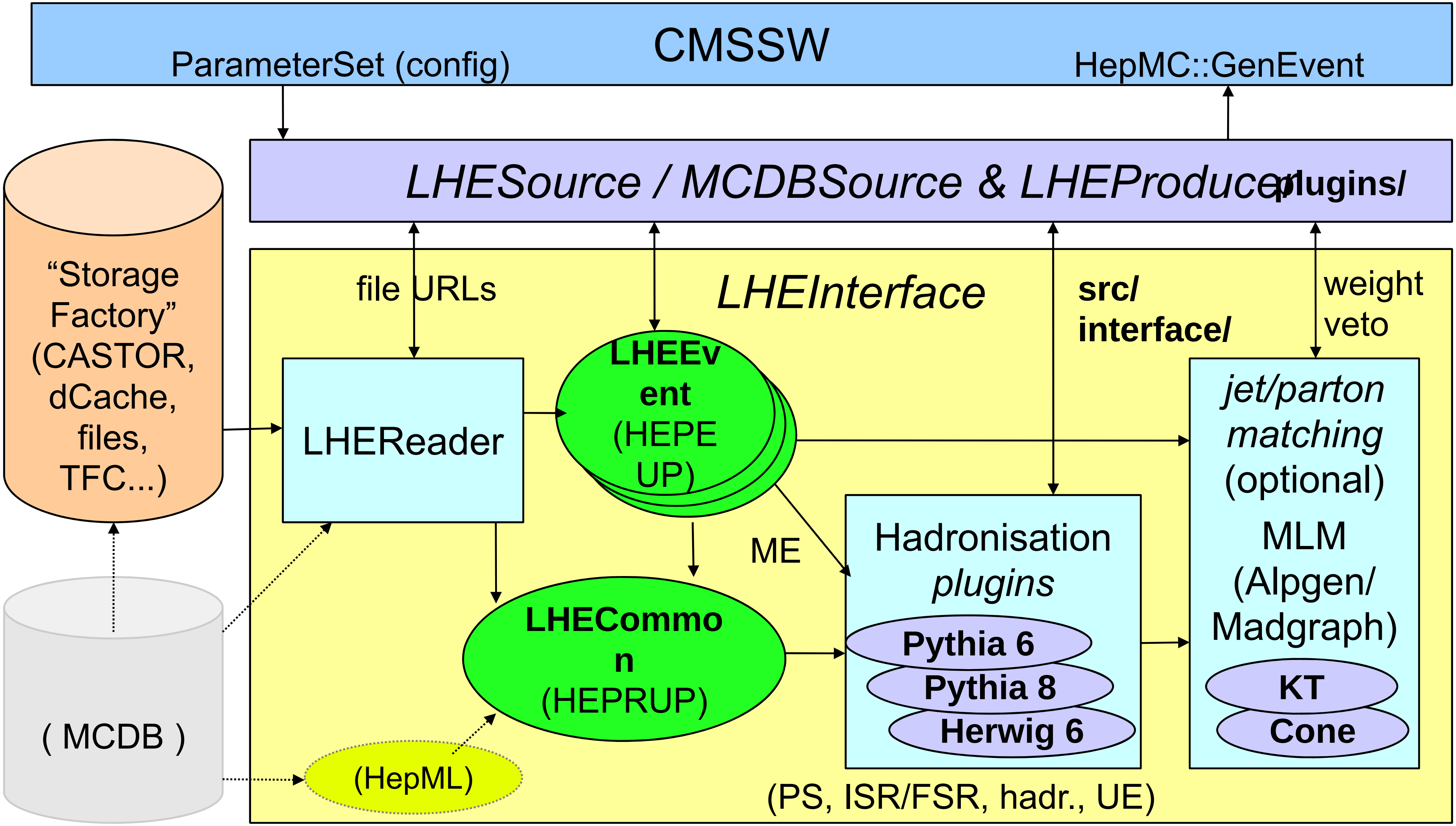


Les Houches Event files

- A lot of event generators only provide **ME** calculations
- Output needs to be fed to **Pythia / Herwig** subsequently
  → common "Les Houches" Fortran **common blocks** defined
- Common blocks only suitable if code is directly glued into executable
- A **common file format** was defined: **LHE files**
  - Allows complete **separation** of **ME production** and subsequent generation chain *(**PS, hadronisation**, ...)*
  - Easier **interchangability** of generators *(e.g. Pythia ↔ Herwig)*
  - Lowers the hurdle for **adoption** of **new** ME generators

- *Another advantage:*
  - Parton-level events are very small *(handy to keep around)*
  - LHE files can be provided by **theorists** independently from experiment *(done so for Spring07 MadGraph production)*

Modular “LHEInterface”



Where does MCDB fit in?

The basic idea:

- First step of ME production is completely decoupled
- Resulting LHE files *(small!)* are uploaded to MCDB
  - “Documentation” of ME generation step *(no throwing away)*
  - Independent of experiment (and experiment software)
  - Can also be provided / validated by **theory colleagues** directly) *(instead of fiddling with integration of code into CMS chain)*

Issues:

- Does not fit well into existing CMS production chain
- Needs new setup for producing ME separately and uploading

MCDB open issues (I)

Issues concerning reading the LHE events (in production)

- MCDB is located **at CERN** - CMS production anywhere on the **Grid**

  → *LHE **data transfer** issue*
- CMS production is done in **chunks of *O(300)*** events

  → assuming LHE file contains 30000 events, this would mean 100 jobs accessing the same file and counting events to find the correct starting point → ***potential I/O bandwith waste***

Possible solutions:

- CMSSW "StorageFactory" supports arbitrary I/O protocols!
  - rfio:// only works locally at CERN, gsiftp:// will be turned off?

    (and srmcp doesn't work behind firewalls)
- **Register** LHE files into **DBS** and use our PhEDEx site replication?
  - Files in DBS are expected to be **EDM** conform *(i.e. **ROOT** format)*
  - Text files aren't **seekable** by event number *(I/O overhead)*

MCDB open issues (II)

Preferred solution:

- Before going into production **convert LHE** files into **EDM** files

- Solution preferred by EDM experts
- **C++ representation** of LHE contents trivial *(both header, per-event and possible additional HepML information)*
- Converter in **both directions** trivial
- Full **information** and production **history** *(book-keeping)* directly in EDM file! *(no loss of information)*
- Per-event exact **reproducability** of event generation step
- **I/O overhead negligible**, ROOT is seekable
- Registration with **DBS** makes it transparent to the system *(and independence from yet another grid transport system)*

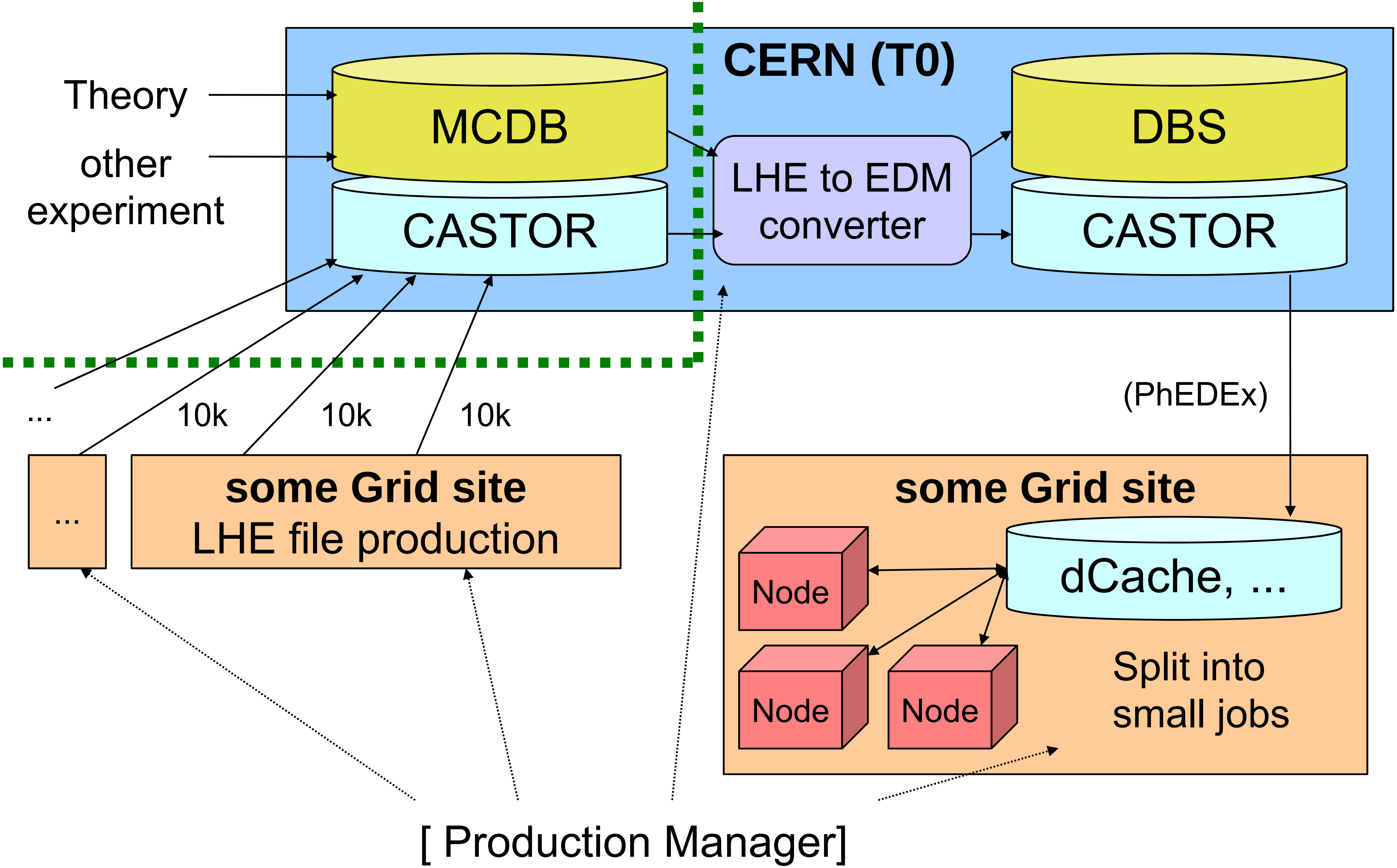
- *Some open framework issues (likely to be solved)*

MCDB open issues (III)

Producing and uploading LHE event data to MCDB

- How is the **authentication** done?
  → grid **certificate** probably sufficient, all CMS production jobs
  run with the VOMS **CMS production role**
- LHE event file production is likely to be done in a **distributed** way on the Grid
  → some sort of automated **“distributed upload mechanism”**
  - Like 100 jobs uploading LHE files belonging to the same **dataset** *(sample)* **in parallel**
  - *Create **MCDB article on the fly** on first upload attempt, merge all other files into same article? (→ to be discussed!)*
  - Possible via **unique identifier** of sample *(like in **DBS**?)*
    → possibility to have an automatic ID string → article ID mapping *(or something similar)* would be perfect

Proposed Architecure



Conclusion & Outlook

- A variety of **generators employing LHE** already in use by CMS
- Plain LHE/MCDB reading already possible *(working!)* for private purposes
- **Generic LHE interface** in preparation

  (probably a good place to start factorization / code sharing)
- A few **technical obstacles** for large-scale **official CMS** production *(mostly on CMS side for the moment)*
  - Integration into **CMS production workflow**
  - **I/O** issues – *prefer a robust solution without adding dependencies*
  - **Distributed LHE file generation** and **upload**
- Complicated, but hopefully feasible **solution proposed**
- **Reusing LHE files** and **proper book-keeping** is a ***must*** for future CMS productions, **MCDB** really is the **most proper way!**

  → *aiming for **integration** into **CMS workflow** before end of **2008!***
- On the MCDB side everything seems to be there
  - Feedback *(and **hands-on** tests)* especially on the upload issue welcome
  - **Evaluation** and **integration** tests are ongoing

Backup Slides

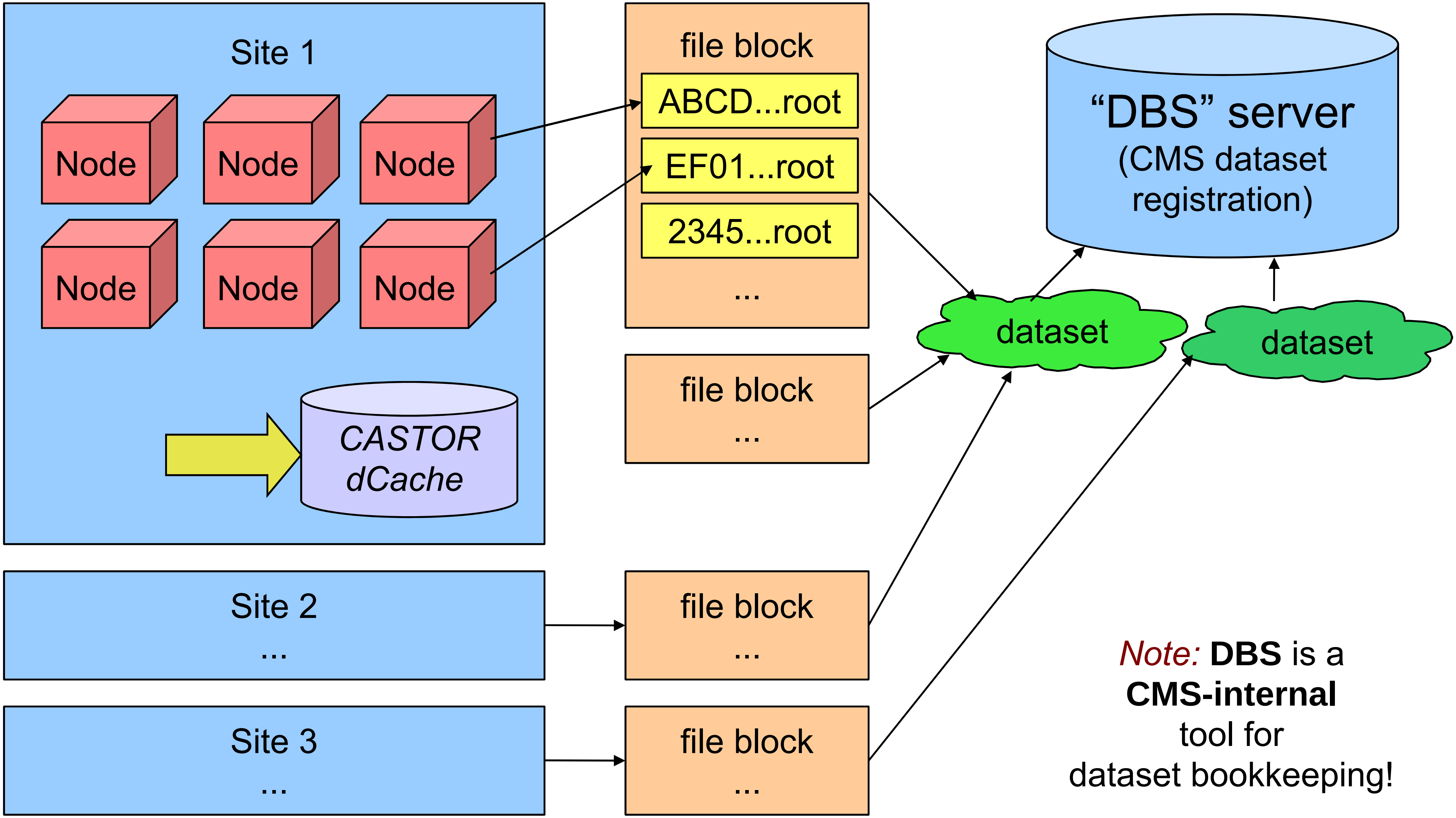
CMS Production Workflow (I)

Official Monte-Carlo production

- decentralised over the GRID (“ProdAgent”)
- Samples divided in “**datasets**”
  - Unique name: /wz2j-alpgen/CMSSW_1_6_7-CSA07-1205907776/RECO
  - → split into *file blocks*

    → file blocks split into *individual EDM files (<GUID>.root)*
  - **one file per cmsRun** *(**O(300) events** each)*
  - Data stored locally (dCache, CASTOR)
  - Local file URLs translation from worldwide “logical filename” using site-local “trivial file catalog” (/store/xxx → rfio://.../xxx)
  - Logical file names registered on **central “DBS server”**
    - File block information: sites which hold the datasets
  - Dataset transfers using “PhEDEx” *(currently based on SRM)*

CMS Production Workflow (II)



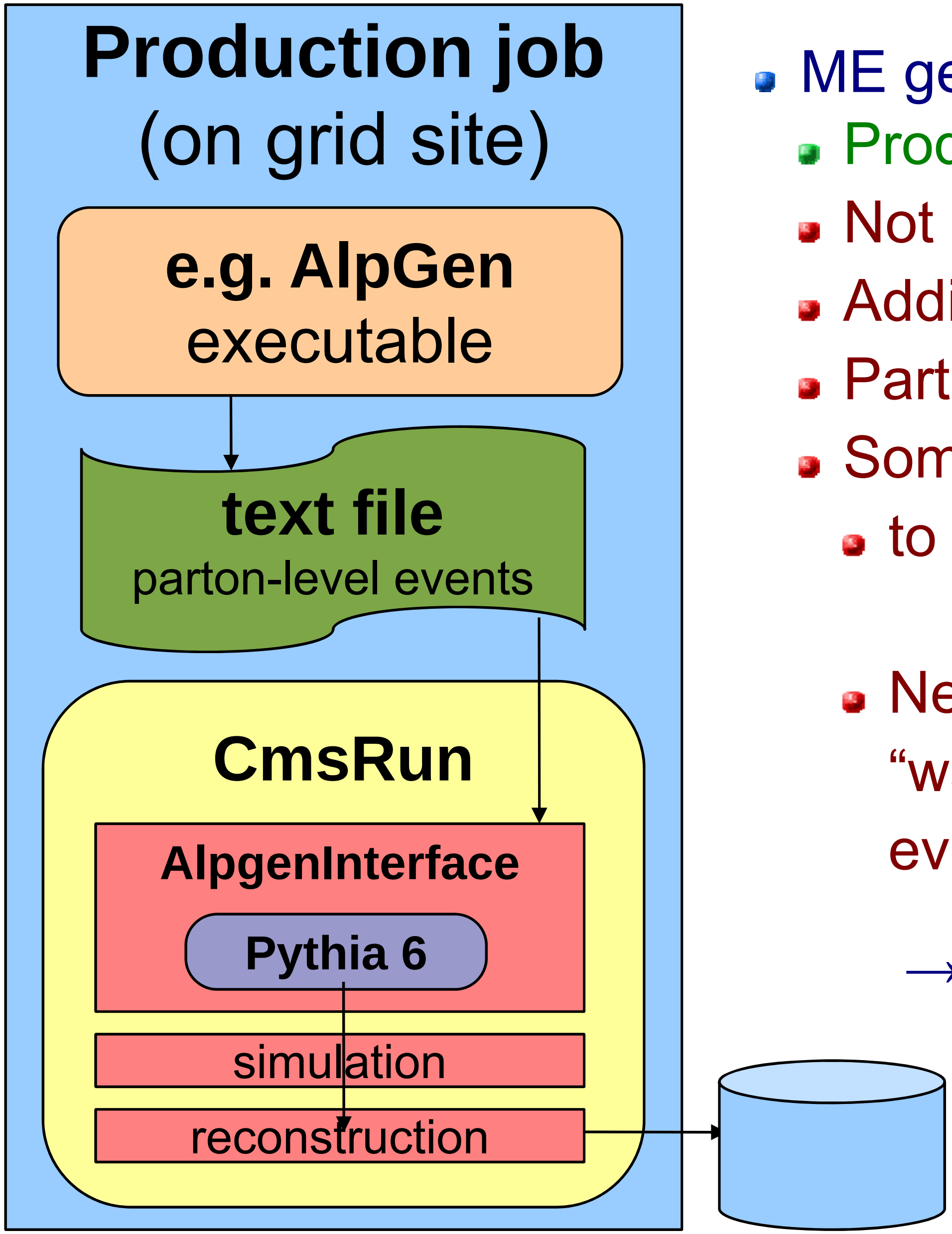
Note: **DBS** is a **CMS-internal** tool for dataset bookkeeping!

Matrix Element generators

- Typical physics event generation consists of roughly three steps
  - **Matrix Element** calculation: the hard process
  - **Parton Shower**: evolution of partons into jets
  - **Hadronisation**: Create final-state particles
  - (also all sorts of radiation and underlying event)
- General-purpose generators like **Pythia** and **Herwig** provide all three steps together, but
  - their **Matrix Elements** are only leading order (LO)
  - Almost the only generators with **PS / hadronisation** models
- A lot of alternate generators exist that provide only **ME**
  - Improved **ME** *(more accurate description of hard emissions)*
  - Other physics processes (SM, SUSY, exotics, ...)

→ *need Pythia/Herwig afterwards to generate full events!*

The CMS approach so far



- ME generator executed directly on-site
  - Production workflow can be kept
  - Not integrated in cmsRun
  - Additional binaries/scripts needed
  - Parton-level files thrown away
  - Some generators *(e.g. MG/ME)* need
    - to be compiled for the process *(per-sample binaries!)*
    - Need preparatory, time-consuming "warm-up"calculations before starting event generation

→ *manual preparation needed anyway!*

(example for one possible generator combination)

Generators status in CMS

Currently available modules

Pythia6Interface
Herwig6Interface
MC@NLOInterface
MadGraphInterface
ALPGENInterface
ExHumeInterface
PomwigInterface
CosmicMuonGenerator
Pythia8Interface
Tauola/Photos
EvtGenInterface
HyjdjetInterface
PyquenInterface
BeamHaloGenerator
ParticleGuns
MCFileReader
CompHepInterface
TopRexInterface
(SherpaInterface)
(Herwig++Interface)

→ *a colourful mixture*

- Pythia used in some, Herwig in others
- Some modules simply read (local) files
- Several modules read generator-specific LHE files (from local disk)

- Considering LHE files a standard, a **simple overall working plain LHE interface** is not officially available
  - *LHE-based interfaces could share LHE interfacing (reader, MCDB)*
  - *Pythia6/8, Herwig(++) interfacing could be factorized (where applicable)*

Planned Productions

The next Monte Carlo production for physics in CMS should bring us to the interpretation of the first data (hopefully).

CMS is currently planning to focus on:

→ Spring08 (April '08) a fast simulation production of the order of ~ 500M events.

→ 3-6 months of data taking at 20% efficiency and 300 Hz storage rate

→ full SM coverage for understanding PD overlaps, trigger tables, training the analyses

→ iCSA08 (May '08) a full simulation production of the order of 100M events, where the main component is QCD+MB. DPG oriented. (simpler than Spring08)

→ mimic the first weeks of data taking with startup simulation conditions

→ test of the computing flow and basic object reconstruction

https://twiki.cern.ch/twiki/bin/view/CMS/DetectorPerformanceMCProduction2008

→ fCSA08 (July '08 if no beam).

→ mimic the first 10-100pb^{-1} of data taking

→ generator plans to be announced, readiness driven by Spring08 + signal MC packages

(from P. Bartalini (NTU), R. Chierici (IPNL-Lyon), CMS software meeting 08.04.08)

Ongoing Production

The 500 million Fast-Sim events that will be produced before the iCSA08 exercise should roughly consist of:

– Min bias	Pythia	100 Mevt
– QCD jets	Madgraph	200 Mevt
– tt + jets	Madgraph	10 Mevt
– t+ jets	Madgraph	2 Mevt
– Photon+jets	Madgraph	25 Mevt
– Z/W + jets	Madgraph	50 Mevt
– Enriched e	Madgraph or Pythia+Filter	25 Mevt
– Enriched μ	Madgraph or Pythia+Filter	25 Mevt
– Enriched γ	Madgraph or Pythia+Filter	10 Mevt
– Bbbar	Madgraph or Pythia+Filter	50 Mevt
– Onia	Pythia	5 Mevt

time scale:
April 08

+ O(1M) fake μ, fake γ

+ O(50M) QCD jets with Pythia (for x-checks). Further smaller Pythia samples?

(from P. Bartalini (NTU), R. Chierici (IPNL-Lyon), CMS software meeting 08.04.08)