Ganga analysis framework for SuperB experiment, work status and plans

Armando Fella on behalf of
SuperB Distributed computing group

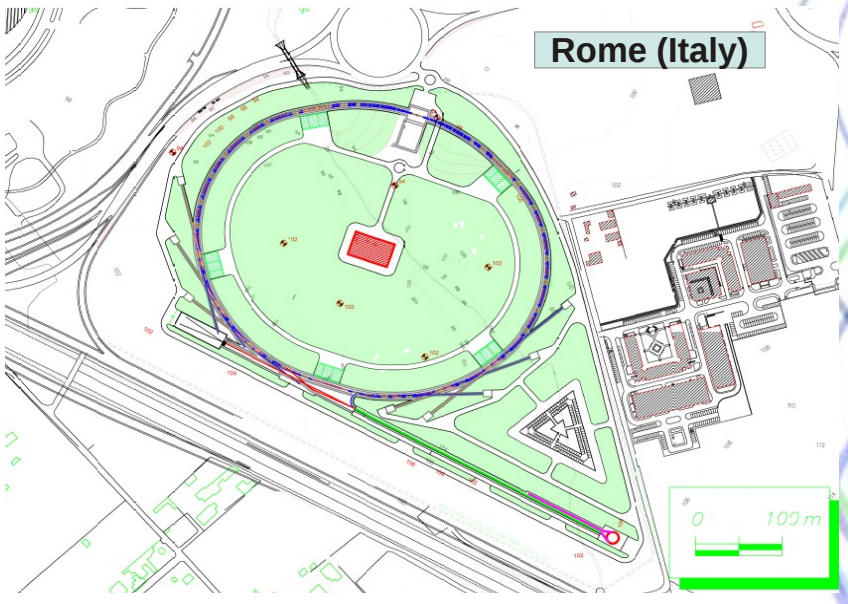
Ganga developer days 8th – 9th February 2012, Birmingham, UK

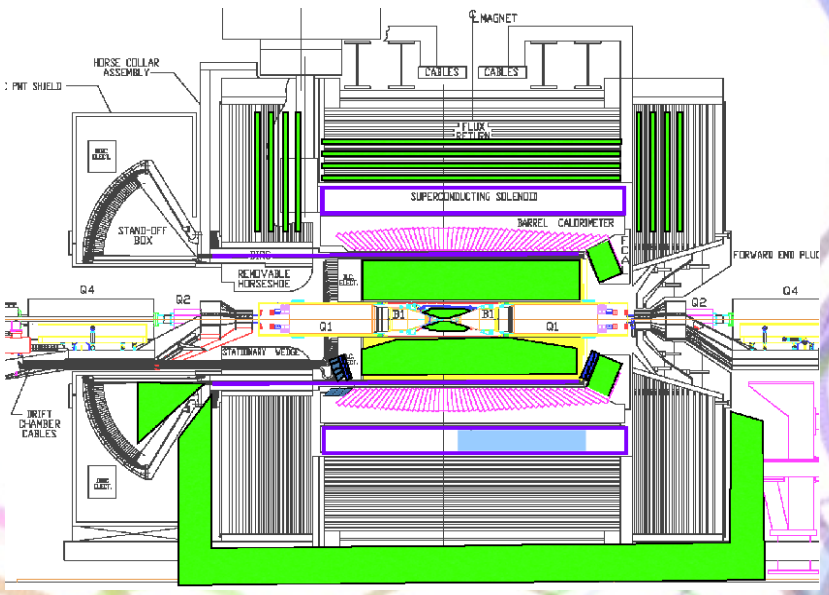
Presentation Layout

- SuperB experiment overview
- SuperB distributed resources
- Analysis framework: design
- Ganga SuperB plugin
- Work planning and conclusion

SuperB experiment

- SuperB is an asymmetric flavour factory with a two-order of magnitudes jump in performance with respect to present B-Factories.
- Data taking is planned to start in 2016
- Computing model definition to be freeze in TDR within one year
  - Luminosity x O(100) w.r.t. present B-Factories (5 years of run)
    - $L_{inst} = 10^{36}\ cm^{-2}s^{-1}$
    - $L_{int} = 75\ ab^{-1}$
  - Flexible parameter choice
  - First level trigger expected rate: ~100KHz
  - Third level trigger expected rate: O(10)KHz
  - Expected event size: ~100KB
- **Large international collaboration:** Canada, Italy, France, Poland, Russia, Spain, The United Kingdom and the United States.





Computing requirement evaluation

- SuperB computing requirement evaluation can be estimated, using as a basis the present experience with BaBar and applying a scaling of about two orders of magnitude
- The following table shows the computing resources needed in a typical year of SuperB data taking assuming a collected luminosity of 15 ab^{-1}
  - Same order as corresponding ATLAS and CMS estimation for 2011

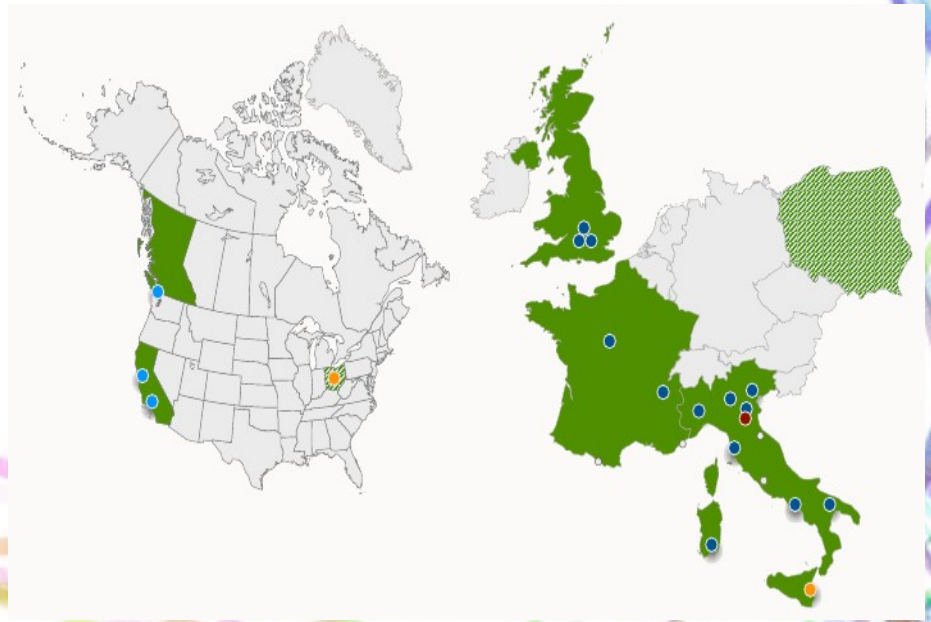
Parameter	typical Year
Luminosity (ab^{-1})	15
Storage (PB)	
Tape	113
Disk	52
CPU (KHep-Spec06)	
Event data reconstruction	210
Skimming	250
Monte Carlo	670
Physics analysis	570
Total	1700

SuperB distributed resources

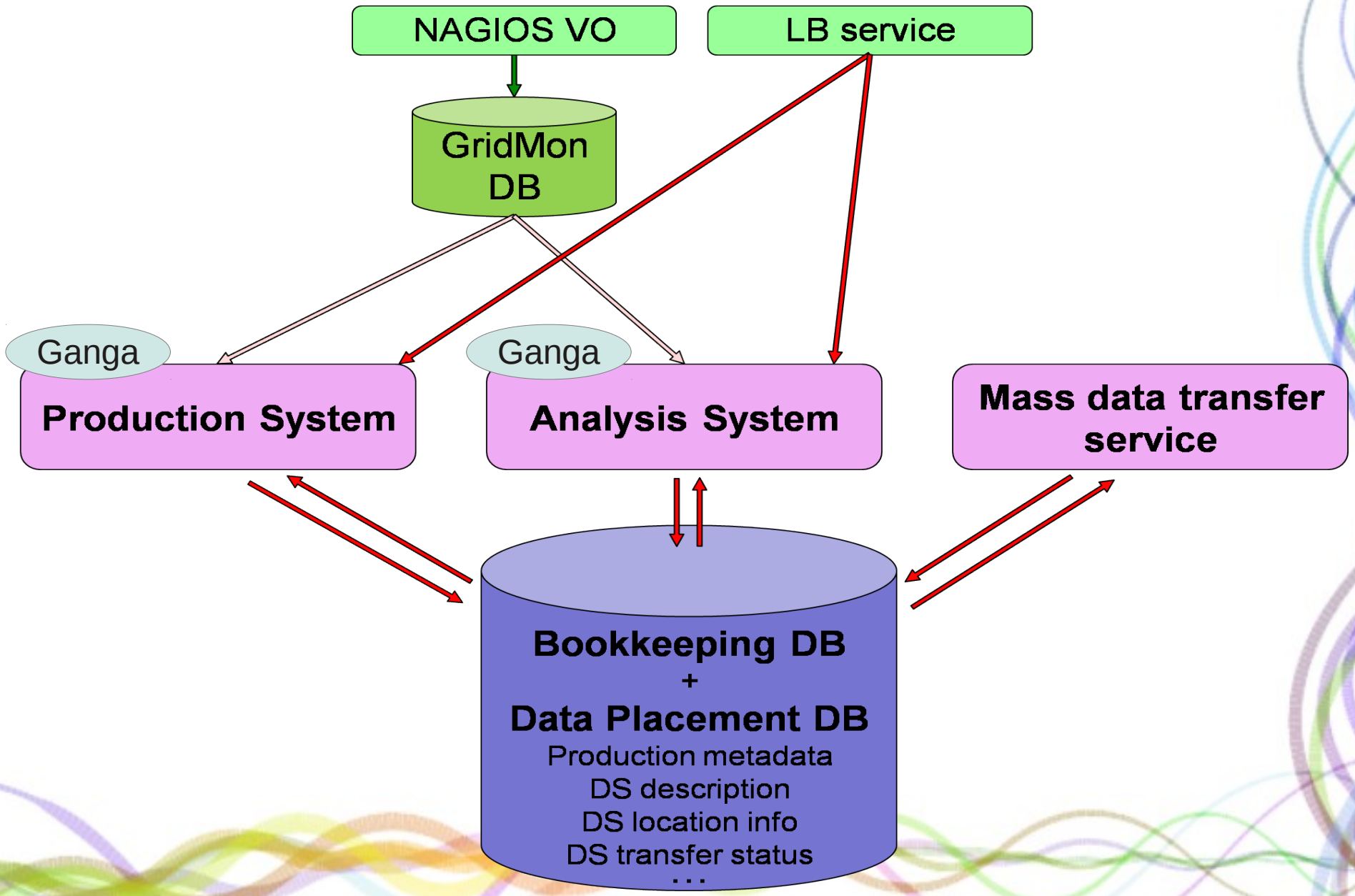
- **LHC Tier-1s** (3)
  - **infn-t1, in2p3-cc, ral-lcg2**
- **LHC Tier-2** (16)
  - **uki-lt2-qmul, uki-southgrid-ralpp, uki-southgrid-ox-hep, grif, in2p3-lpsc, wt2(slac), cit-cms-t2b, victoria-lcg2, cyfronet-lcg2, infn-bari, infn-catania, infn-lnl-2, infn-milano, infn-napoli-atlas, infn-pisa, infn-torino**
- **Other** (8)
  - **infn-ferrara, infn-perugia, infn-cagliari, napoli-grisu, napoli-unina, in2p3-ires, cit-hep-ce, osc**

Green: EGI sites
Red: OSG sites

Site	Min (cores)	Max (cores)	Disk (TB)	SRM layer	Grid Org.
RAL(T1)	200	1000	25	Castor	EGI
Ralpp	50	500	5	dCache	EGI
Queen Mary	300	3456	120	StoRM	EGI
Oxford Univ.	50	200	1	DPM	EGI
IN2P3-CC(T1)	500	1000	10	dCache	EGI
Grif	50	300	2	DPM	EGI
in2p3-lpsc	50	100	2	DPM	EGI
in2p3-ires	50	100	2	DPM	EGI
CNAF(T1)	500	1000	127	StoRM	EGI
Pisa	50	500	0.5	StoRM	EGI
Legnaro	50	100	1	StoRM	EGI
Napoli-infn	50	100	5	DPM	EGI
Napoli-grisu	50	300	5	DPM	EGI
Napoli-unina	50	300	5	DPM	EGI
Bari	160	260	0.5	StoRM/Lustre	EGI
Ferrara	10	50	0.5	StoRM	EGI
Cagliari	10	50	1	StoRM	EGI
Perugia	10	50	1	StoRM	EGI
Torino	50	100	2	DPM	EGI
Frascati	30	100	2	DPM	EGI
Milano	50	100	2	StoRM	EGI
Catania	?	?	?	StoRM	EGI
Slac	400	400	10	NFS	OSG
Caltech	200	400	4.5	NFS	OSG
OhioSC	?	?	?	dCache	OSG
Victoria	50	100	5	dCache	EGI
Cyfronet	100	500	10	DPM	EGI
Total	3070	11066	349		



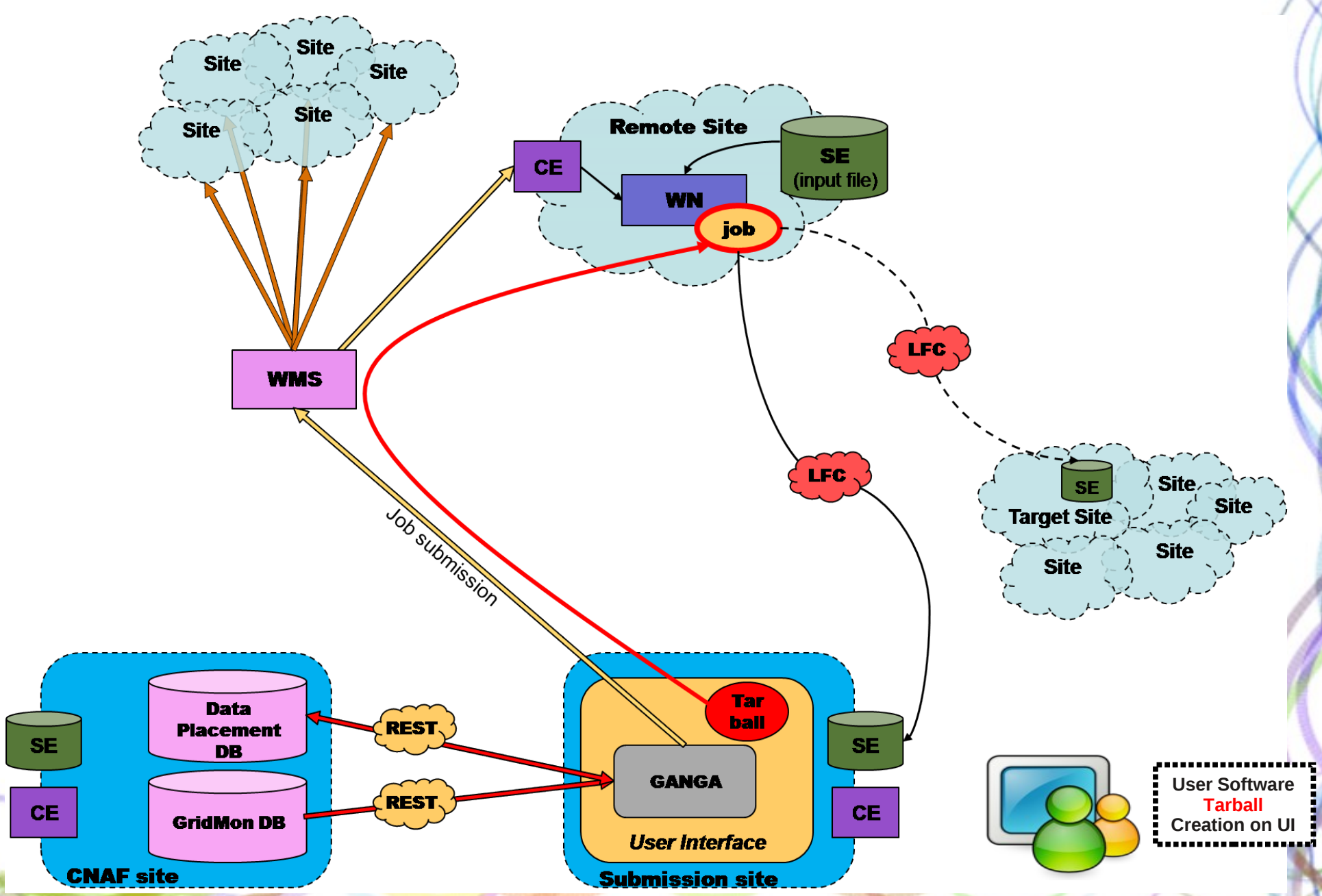
High level scenario



Analysis use case, general info

- We use Ganga for job preparation, monitor and submission
  - Ganga, SuperB job wrapper and DB do all the work
- Job data management
  - data driven paradigm --> ongoing R&D for pure WAN input data access scenario (two years passing through mixing infrastructure)
  - Input files max size: 10GB, Output files max size 3GB, max job RAM consumption 1GB
  - Job stage in/out transfers performed via Lcg-Utils --> ongoing R&D for direct access to specific storage system SRM2.2 compliant
- Authentication, Authorization: VOMS, Role driven authentication
- File catalog: LFC
- WMS submission to exploit Grid flavour interoperability features
  - Direct submission to site's CE

Analysis job workflow



Analysis use case step by step

- **Before accessing ganga**
  - User prepares his sw directory containing executable and configuration files needed for analysis or simulation production purpose
    - Executable should follow few conventions about input file path and location for output files
- **Job preparation from ganga interface**
  - User can select the input dataset, declare the evt to be processed per job, optionally select run and output site, create the output dataset, define executable path, finally submit the job
- **Job run time and stage in procedure**
  - SuperB job wrapper untar user sw package, check environment, transfer requested input files to the conventional area on wn via lcg-cp and launch the user executable
- **Stage out**
  - At job completion, all the output files residing into the conventional area will be transferred and registered in LFC to output site (default: submission site, via lcg-cr). File containing the list of LFN output files is transferred via outputsandbox. Job wrapper is able to insert job status in DB.
- **Merging step** (to be designed)
- **Data movement**
  - User can perform data transfer of its own dataset via a set of specific methods implemented into SuperB ganga plugin. Not intended to be mass data transfer

Ganga site setup

- One .ini file per site
- No need of .gangarc (do we need to prevent user to write its own .gangarc, how to manage it better?)
- One simple Ganga wrapper launcher per site
  - ${SW_AREA}/ganga/install/${GANGA_REL}/bin/ganga --config-path=${CONFIG_INI_PATH}/analysis_superB.ini -o[Configuration]gangadir=${HOME}/gangadir
- Ganga distributed with experiment software via CVMFS (in short term)
- SuperB job wrapper distributed with Ganga plugin

SuperB Ganga plugin I

- **Use-case driven procedure**

```
+----+-----------------------------------------------+
| id | work_session_type                             |
+----+-----------------------------------------------+
| 0  | FastSim Production Analysis                   |
| 1  | FullSim Production Analysis                   |
| 2  | Personal Production (Fast or Full)            |
| 3  | Personal Analysis                             |
+----+-----------------------------------------------+
```

- **Fast/Full simulation ntuples reduction or analysis**: (testing)
  - **IN** dataset --> **OUT** dataset, bulk submission
- **Personal simulation production**: (testing)
  - **IN** null --> **OUT** dataset, bulk submission
- **Personal analysis**: (testing)
  - **IN** dataset --> **OUT** dataset and/or not structured files, bulk submission

SuperB Ganga plugin

Package plugin

source code

Application definition
Use case definition
Dataset management
DB http interface

Submodules

- plugin.BOOT
- plugin.Lib: *General information and conventions about job data management*
  - plugin.Lib.SuperBApp: *SuperB setup of application and RTHandler Ganga methods*
  - plugin.Lib.SuperBBulkSubmission: *Define number of subjobs per use case*
  - plugin.Lib.SuperBDatasetManager: *Unique class to monitor and transfer Dataset*
  - plugin.Lib.SuperBHelp
  - plugin.Lib.SuperBInputDatasetAnalysisAnalysis: *Manage the input dataset job selection for the use case analysis*
  - plugin.Lib.SuperBInputDatasetProduction: *Manage backgroud frame job input dataset for use case personal production*
  - plugin.Lib.SuperBInputDatasetProductionAnalysis: *Manage job Input dataset coming from official and personal productions*
  - plugin.Lib.SuperBOutputDataset: *This class register output dataset to DB.*
  - plugin.Lib.SuperBRequirements: *Get site status and host name info from gridmon DB and build up jdl requirements.*
  - plugin.Lib.http_interface: *Implemantation of HTTP interface to DB resource.*
  - plugin.Lib.json: *JSON (JavaScript Object Notation) <http://json.org> is a subset of JavaScript syntax (ECMA-262 3rd edition) used as a lightweight data interchange format.*
    - plugin.Lib.json.decoder: *Implementation of JSONDecoder*
    - plugin.Lib.json.encoder: *Implementation of JSONEncoder*
    - plugin.Lib.json.scanner: *JSON token scanner*
    - plugin.Lib.json.tool: *Command-line tool to validate and pretty-print JSON*
  - plugin.Lib.objectid: *tool suite to generate ObjectIds to as unique key for dataset creation in DB*
  - plugin.Lib.utils: *Utility library*
- plugin.PACKAGE: *Refer to Ganga/PACKAGE.py for details on the purpose of this module.*

EpyDoc: http://bbr-serv09.cr.cnaf.infn.it:8080/doc/ganga/

Simulated event analysis use case

Now it doesn't work due to class refactoring

- j=Job()
- j.name = 'myJob'
- j.application = SuperBApp()
- j.application.setOutputSite()
- j.application.exepath = 'analysisExe.sh'
- j.application.sw_directory = '/users/ganga_util/GangaSuperB/test/analysisSoftware'
- j.inputdata = SuperBDataset()
- j.inputdata.setRunSite()
- j.inputdata.addInputDataset()
- j.inputdata.createJobInput()
- j.inputdata.setOutputDataset()
- j.outputsandbox = ['graphs']
- j.splitter = SuperBBulkSubmission()
- j.merger = TextMerger()
- j.merger.files = ['severus.log', 'output_files.txt']
- j.merger.ignorefailed = True
- j.merger.compress = True
- j.backend=LCG()
- j.submit()

http://mailman.fe.infn.it/superbwiki/index.php/Ganga_setup_for_SuperB#Hands_on.2C_guided_procedure

Personal Production use case

```
j=Job()
j.name = 'Elisa'
j.application = SuperBApp()
j.application.setOutputSite() #default is submission site
j.application.sw_directory = '/storage/gpfs_superb/users/eisa/FastSim/V0.3.1_test/'
j.application.exepath = 'workdir/pacBtoKstar0NuNu-dg4-3-ganga.csh'
j.inputdata = SuperBInputDatasetProduction()
# answer Yes:   include bkgframe mixing with evt simulation
j.inputdata.findSoftwareVersion()
j.inputdata.addRunSite()
j.splitter = SuperBBulkSubmission()
j.merger = TextMerger()
j.merger.files = ['severus.log', 'output_files.txt']
j.merger.ignorefailed = True
j.merger.compress = True
j.backend=LCG()
j.submit()
```

Monitoring

- We are using the Ganga Task Monitor: http://gangamon.cern.ch/ganga/
- We are not using GangaGUI at present time
  - We are interested in its future development plan
  - Ganga WebGUI status?
- We are very interested in Ganga checker functionality
  - see “LHCb Summary and Future Plans” by A.Richards

Future work

- Mysql to Postgres information system porting is on course
  - Class refactoring
- Job experiment specific status != grid status ---> to Ganga
- Merge step implementation
- OSG site exploitation, Condor backend
- Task features integration, resubmission
- Prepared state integration
- We have to release the first working Ganga system to the physics community within April 2012
  - The main functionality should work properly within such a dead line. Our desiderata is to include the merging step and job specific status, in some way, implementation
    - Testing, validating, user training session

Issues, questions

- Ganga inputsandbox prestage policy:
  - who clean up SE areas? When? What SE access point will be chosen? Is it disabling?
- Merge step
  - Need to understand the best way to proceed in SuperB scenario. Future Ganga development?
- DB HTTP/REST interface implemented in Ganga?
- Intermittent slowness j.submit execution
- Open LHCB or ATLAS documantation, impossible?
  - Ganga Cookbook as best practice container? WOW YES!
- In general we need feedback to improve, correct SuperB design and implementation.
  - What we do not use, but it's incredibly useful?
  - better data structure, better core class interaction, better class design, fail over mechanism in ganga

Thanks all and backup slides

SuperB Ganga plugin II

- Job Input data management: data driven model
  - Class SuperBDatasetX, the methods (under refactoring):
    - addInputDataset(): shows all datasets, asks user to choose one, and add it to the dataset list to be analyzed, the results can be filtered per production and/or per site if previously set.
    - delInputDataset(): delete one dataset to the dataset list to be analyzed.
    - setProduction(): filter on a specific simulation production the shown datasets by addInputDataset
    - setRunSite(): select a the job destination site. Work as filter for addInputDataset in showing datasets list residing on such a site.
    - createJobInput(): get the whole lfns of the dataset list, divide it according to the user's requests and the input size limit then create some txt list files that will be used by the subjobs.
    - setOutputDataset(): define the output dataset name; initialize related metadata on data placement DB

SuperB Ganga plugin III

- User will provide to ganga a tarball containing executable, config files and eventually a directory containing the files of lists of job input files
  - Class SuperBApp, the methods (under refactoring):
    - OutputSite(): shows all the supported outputSites, asks user to choose one site then sets target site where job analysis output will be transfered.
  - Properties:
    - exepath: executable *relative* path after analysis software unpacking.
    - sw_archive: Local UI absolute path to analysis software compressed archive.
    - sw_directory: Local UI absolute path to analysis software directory. eg:
    - outputSite: target site for job stage out. eg: INFN-T1.

SuperB Ganga plugin IV

- Use case Personal Production

.

  - Class SuperBDatasetPersonalProduction, the methods:

    - Opt, addRunSite(): add a site where the job will be routed.
    - Opt, delRunSite(): del a site where the job will be routed.
    - Opt, setOutputDataset(): define the output dataset name; initialize related metadata on data placement DB