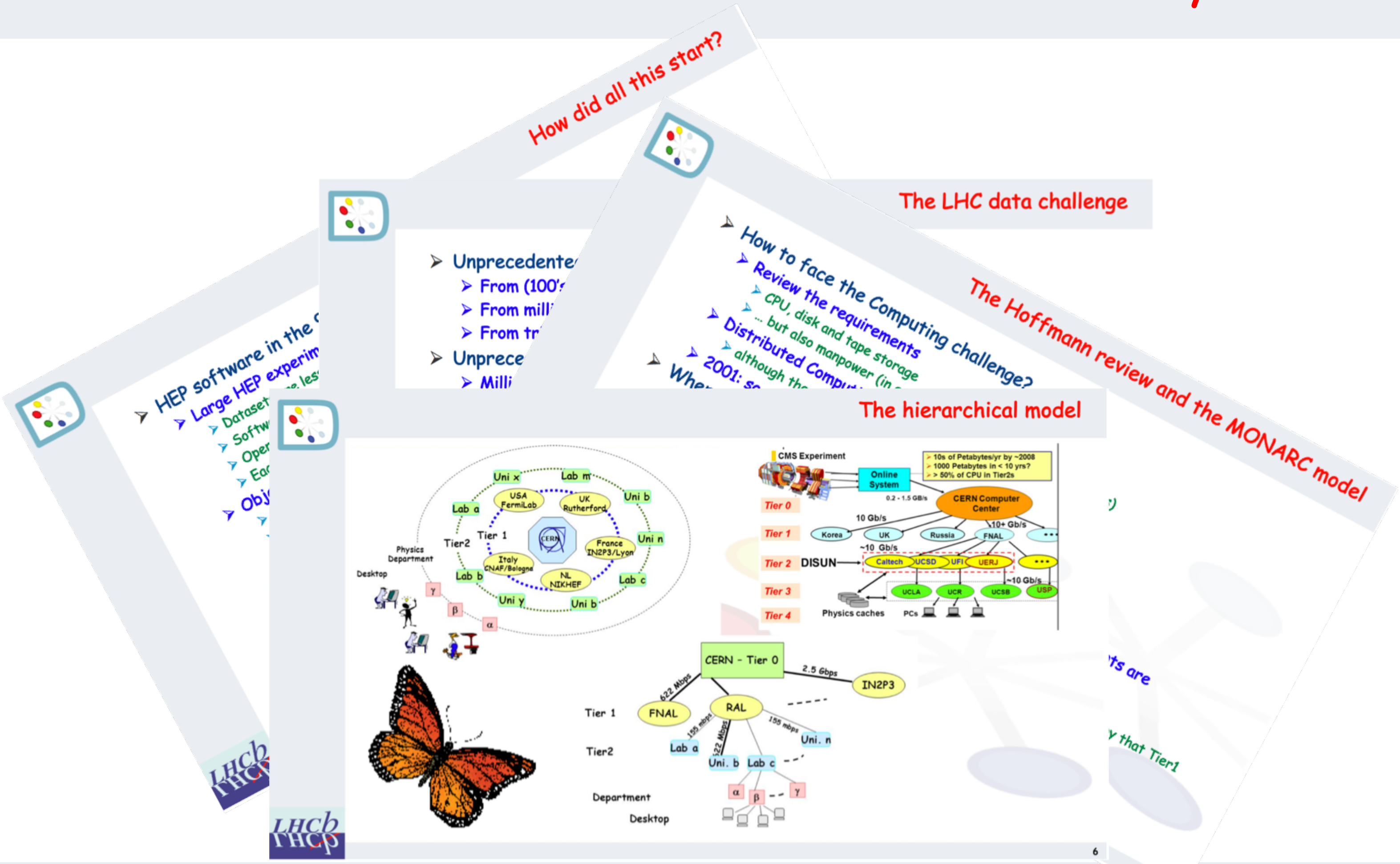
A brief history of LHC Computing

Philippe Charpentier
CERN

Disclaimer

- This presentation is neither
  - An official historical review of LHC computing, nor
  - An exhaustive review of all events that happened in LHC computing
- It is more
  - A testimony from my personal experience in the past 18 years
  - An overview of the events that I personally believe have marked those 18 years
- It is also
  - Strongly biased by my personal experience
  - A collection of subjective feelings
- Hence
  - No (or few) names, no precise dates…

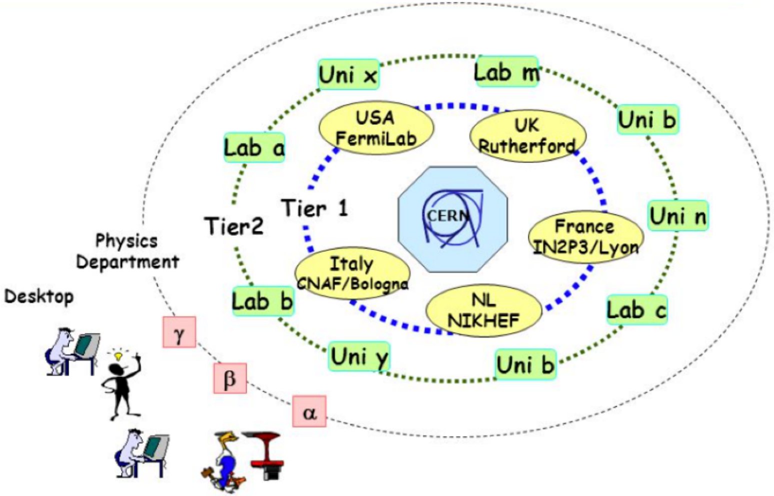
My first attempt

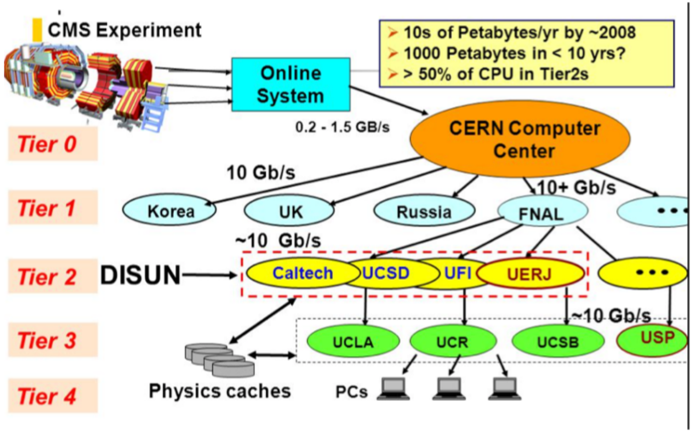


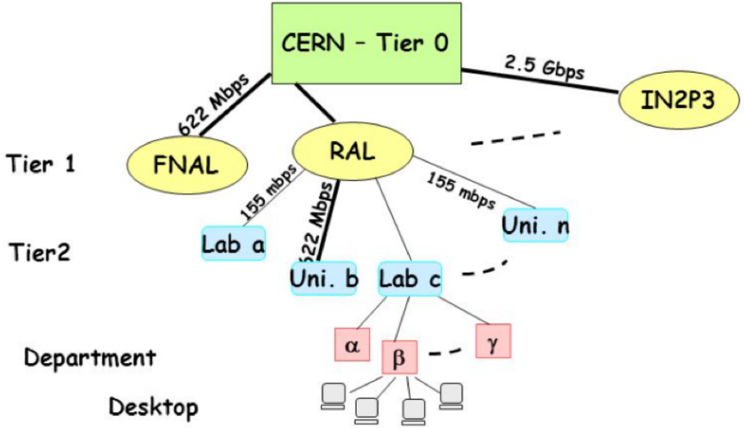
My second attempt

- After 4 slides, I hadn't even left the past millennium
- I only have 20 minutes… and you would be bored with an exhaustive history
- … so, let's be less precise and exhaustive and give an overview…
- Once upon a time, the LHC and its experiments were being built… but computing had not really been considered at the required level…

The hierarchical model: MONARC (1998)







The Grid paradigm

- A way to implement a distributed computing paradigm

- Was proposed by I.Foster and C.Kesselmann in 1998
  - A set of interconnected compute and storage resources
  - Deploy "middleware" that allows a seamless access to these resources
  - In the end: looks like a single very large computing center…

- The idea was picked up in 2000 (Padova CHEP) for LHC computing
- Some middleware existed with the Globus project
  - But more was needed for the LHC computing
  - Launching an R&D program sponsored by the EU: European DataGrid
    - Based on existing Globus middleware
    - Focused on a set of work-packages… and on people
    - No real requirements' document, no architecture
    - … but this was R&D!

Getting organised for LHC Computing

- Creation of a "Collaboration" for coordinating LHC Computing
  - LHC Computing Grid (LCG) in the early 2000
    - The 5th LHC collaboration (i.e. followed by the LHCC)
    - Many initial governing bodies (PEB, GDB, SC2, OB…)
  - More a coordination body than a "Grid initiative"
  - Several "areas"
    - Middleware area
      - No development proper (done in EDG and then EGEE)
    - Deployment area
      - Try and coordinate sites
    - Fabric area
      - for CERN fabric only
    - Applications area – not part of the Grid proper
      - Architects Forum
- LCG will then extend to integrate non-European sites
  - Became Worldwide (WLCG)
    - Collaboration Board, Management Board, Grid Deployment Board, Overview Board

- The Grid is nothing but a huge computing centre… just fine!
  - Users should submit their jobs as if it were a batch system
  - Data storage should look like a huge (although redundant) storage
  - "The Grid is to data processing what the Web is to information"
- … but …

- Each site is allowed to make its own decisions concerning
  - Its internal batch system (should also serve other users)
  - Its internal storage system
  - However quite difficult to deploy Grid middleware initially
- How can central brokering decisions be made?
  - For compute resources: it should know at any time the exact status of the whole system
  - For storage resources: it should know where data are and how to access them

- Information transmission cannot be instantaneous
  - Hence a central service cannot know precisely the status of the whole Grid
    - Snapshots only every few minutes
    - Many things can happen that invalidate the decisions made
  - How can one gather this information?
    - Resources broadcast their status to a central service
    - Decision making services (Resource Broker) interrogates this information repository
  - This architecture doesn't scale…
- Initial Resource Brokers (RB) do not scale above a few 1000 jobs
  - System should sustain 100'000's if not millions
  - Proposed solution: use many of them
    - But they don't know about each other, hence the same decisions at the same time
      - … which are then wrong!
  - Computing Elements (CE) necessary at each site
    - Front-ends of batch systems (several CEs needed for large sites)
    - Don't scale either, don't offer all required functionality

What about requirements and architecture?

- In the early 2000's, developments had started (EDG, EGEE) without
  - A requirements' document
  - An architecture document
- In October 2003, a document was produced called HEPCAL
  - High Energy Physics Common Application Layer (!)
  - Written jointly by all 4 experiments, "loose canons" and "Grid experts"
  - Updated in March 2004
  - This document was very interesting but was never really considered by middleware developers
- In June 2004, another document was produced as a roadmap
  - ARDA: A Roadmap for Distributed Analysis
  - Architecture, based on HEPCAL requirements, and implementation proposals
    - Architecture based on AliEn and Dirac: services and agents
  - This document was even more interesting but was never considered either
- In the end, the current implementations fulfil the HEPCAL requirements and their architecture is deeply inspired by the ARDA architecture

Some more expectations…

- Data Management: replica catalogs needed to keep track of where files are
  - Files may have several replicas for redundancy and easy access from jobs
  - Globus had developed a File Catalog
    - … but it also fell apart above a few 1000 files!
  - Emergency need for a replacement
    - Within a few weeks, the Castor-1 name-server was converted into a file catalog!
    - The LCG File Catalog (LFC)
      - It will in the end be used by experiments for more than 10 years!
- Storage systems offer different service classes
  - The "Mumbai ontology"
    - At WLCG workshop before CHEP 2006
    - Experiments: need to express different storage classes:
      - Latency (tape, disk…)
      - Quality (resilient, temporary…)
    - T0D1, T1D0, T1D1…
    - Proposed solution: SRM (Storage Resource Management)!
      - This was at the time an abstraction of storage management under definition
      - Agreement to work on (developers) and to use (experiments) this abstraction

… with various levels of success …

- LFC and SRM were globally a success!
- LFC
  - Was robust and it reasonably scaled
  - Used differently by different experiments
    - Central catalog
      - … and its variant of replicated services for redundancy and scalability (LHCb)
    - Distributed catalog (one per site)
      - … not easy to have a global view (ATLAS)
- SRM
  - The main limitation was the adaptation to existing storage solutions
  - Castor, dCache, DPM (another spin-off of Castor-1)
    - Each had its own interpretation of the specifications!
    - … and its own implementation
  - Not all functionalities were implemented
    - Agreement between experiments (mostly ATLAS and LHCb) on common features
  - Quite successful (as still in use)… although (too) heavy on several aspects
  - Still the only known effective way of handling tapes and service classes

MONARC adapted to the Grid

- MONARC describes a hierarchy of site roles
  - Tier0: where data comes from and is first reconstructed
  - Tier1: national centres, meant for running simulation and for real data reprocessing
  - Tier2: regional centres, meant for analysis
- The LHC Computing Grid uses a similar hierarchy
  - Tier0 (CERN): main repository for real data and its reconstruction
  - Tier1: large (national) centres, with custodial storage (i.e. tape) and high level of availability
  - Tier2: smaller (regional) centres, with only disk storage, getting their data from an associated Tier1. Lower requirements in terms of reliability and availability
- Although the names (TierX) are the same, the scope and meaning is different…
  - Sites' roles are *in fine* defined by the experiment's computing model
  - At the beginning ATLAS and CMS were meant to follow MONARC's prescriptions
  - From the beginning LHCb defined different roles (which was "criticised")
    - Real data reconstruction at Tier0 AND Tier1s
    - Analysis at Tier1s, simulation at Tier2s

Workload Management: hopes and facts

- This was the most critical part (limitations of RBs)
- European initiative for middleware development
  - Enabling Grids for E-sciencE
    - Last E used to be Europe… but was then much broader scope…
  - Continued on the same paradigms as EDG
    - RB replaced with WMS (Workload Management System)
    - LCG-CE replaced with CREAM (for batch abstraction)
  - Same paradigm, same effects…
    - More scalable than previous generation
    - … but still unable to know the wave function of the Grid!
- Limitation of central WMS decision
  - Wrong decisions lead to bad sharing between sites
  - Possible solution is for users to broker the site themselves: no decision to make!
    - … but then why have a complex system if to submit to a selected (set of) CEs
- This was used for a long time by several VOs… however… changes were to come…

The pull-push change of paradigm

- If making a central decision as to where to run a job is so difficult…
  - … change paradigm!
- Do not send jobs to sites any longer (push), keep them in a central queue (VO-dependent)

- Instead, sites fetch jobs in the central queue (pull)!
  - "Only" needs to send "pilots" on the site whose role is to fetch jobs
    - Analogy with pilot fish
  - Can then implement policies at the central queue level
    - Set priorities (internal to the VO)
    - Select for each job a set of eligible sites: only pilots from that site will fetch them
- Main advantages
  - Delay matching: no risk of bad decision, as the resource is booked by the pilot
    - Pilot jobs represent a resource overlay
  - Can implement any of (user-defined) site capability criteria

The pilot-job paradigm

- Was first implemented by LHCb and ALICE (around 2002)
  - DIRAC and AliEn
- Immediately demonstrated a higher success rate
  - Resources are used more efficiently
  - Pilots can even run more than one "job", if resources allow
- Allows the VO to set relative priorities to jobs in the queue
- It took however several years before pilots were used by all LHC experiments
  - ATLAS initiated in 2006 (CHEP Mumbai) with Panda
  - CMS started to use WMS-GlideIns (from HTCondor) a few years later
- I believe that today <u>ALL</u> LHC jobs on the Grid run through pilots
  - WMS was decommissioned, CEs largely simplified (ARC, HTCondor rather than CREAM)

Pilot out of the vacuum?

- With the pilot paradigm, the only interaction with the sites consists of deploying pilots!
- Natural evolution of job submission:
  - Submit pilots when jobs are available for a given (set of) site(s) in the central task queue
  - Caveat: jobs are not necessarily executed by "their" pilot (i.e. the pilot whose submission it triggered)
    - Some pilots may not fetch any job
  - It is also a rather heavy process
- The Vacuum pilot factory (VAC, VCycle)
  - Submit pilots on behalf of a (set of) VO(s), at a defined pace
  - Pilots report when they can't fetch jobs
    - Then the pace of submission is decreased… and vice-versa
  - This paradigm allows an easy implementation of fair share
    - Simply adjust the pace for various VOs
  - This is applicable for Virtual Machines, containers, batch system pilot jobs

What happened to the hierarchy of sites?

- LHCb had a completely different model from the start
  - CERN was always used as any other Grid site (i.e. through pilot jobs etc…)
  - Analysis running at Tier1s and CERN using a pure Grid paradigm
    - No national preference, pure resource sharing
  - Simulation running at all sites with lower priority (back-filling resources)
- Other VOs consider CERN/Tier0 as a special case, analysis at Tier2s only (MONARC!)
  - CERN not used as a Grid site, but using local batch system submission
- With time, the hierarchical roles diluted
  - Tier2s no longer directly associated to a Tier1
    - May get / upload their data to a set of other sites
  - Tier1s no longer "uncorrelated"
    - Need to exchange data between Tier1s in addition to Tier0<->Tier1
    - Consequence: create a full network connectivity between Tier0 and all Tier1s (LHCOPN)
  - Still for long some VOs dedicated Tier2s to analysis and analysis to Tier2s
    - Even using the notion of "national" computing, i.e. reserve or privilege nationals for running at a site
- Now
  - Roles have almost completely diluted: any type of job may run anywhere
    - Job brokering still using data location, but no reserved role for sites
  - Full mesh networking including Tier2s (LHCONE)

Recent trends

- Limit the number of sites with custodial storage (a.k.a. tape)
  - Will be hard to achieve, as sites' infrastructure exist
  - … however a few sites with tape would be enough (a few per experiment)
- More use of networks
  - Using *xrootd*, data can be accessed over the WAN
    - No strict need to run jobs where data are
    - Data Storage Federations
  - Several ways to implement federation of storage
    - Using continental / national xrootd redirectors (ATLAS, CMS)
    - Using replica catalog information to instruct jobs (LHCb, ALICE?)
  - In theory (with infinite network bandwidth) one could get rid of job brokering
    - Allow any job to run anywhere
  - In practice, for efficiency, federations mostly used for failure recovery
    - Still broker jobs where data is resident and online
    - If files are not available, or not accessible, use other replicas over the WAN
    - 5 to 10% of file access are over the WAN

What about commonalities?

- Each experiment now has its own ~~Grid access~~ Distributed Computing System
  - Because this is how to implement the experiment's Computing Model
  - Because generic tools are not necessarily adequate
  - Because generic tools were not delivering the expected (level of) service
  - Some systems have integrated DMS and WMS (ALICE with Alien, LHCb with Dirac), some have separate DMS and WMS (in ATLAS and CMS).

- Common services left
  - File Transfer System (FTS): very successful generic transfer service
    - Successful as the requirement is "simple": transfer this file from A to B
  - Computing Elements
    - Large simplification in the recent past
    - Recommendation to use simple / integrated CE (HTCondor-CE, ARC CE)
  - VOMS for authentication
  - De-facto standardisation on *xrootd* for file access (not a Grid development)
  - Software distribution using CVMFS (not a Grid development either)

… and what next?

- Applications tend to depart from classical sequential single-threaded ones
  - Because processors become more and more powerful, but due to their multiple cores
  - Memory is already an issue and will become more and more an issue
    - Large part of the cost of WNs
  - Multi-process and multi-threading applications are more and more used and will develop
- Applications are being optimized to use vectorization heavily
  - This may cause problems of compatibility in the future
    - Problem for the experiments: maintain code and binaries for multiple hardware solutions
    - Already now SSE4 for example is not present on all WNs
- Accelerators and coprocessors are being considered by experiments
  - Is it a hype or is it something that will be actually useful to help solving the HL-LHC challenge and earlier LHCb and ALICE upgrade challenges?
    - Not clear to me whether it is desirable / usable outside special applications (trigger in particular)
- Will the funding agencies continue to fund academic computing centres?
  - Economically worth buying computing resources to large companies (on clouds)
  - How shall we use clouds?

Using commercial or private clouds

- Running private Virtual Machines on commercial clouds is now easy and cheap
  - … but is it the future?
  - For example CernVM can be easily customized and used as a pilot
  - Containers are more and more favoured (much lighter, easier to deploy…)
  - Different points of view concerning usage of clouds
    - Using directly cloud resources by the experiment (i.e. the experiment deploys VMs / containers)
      - This is the most elastic solution: resources are used (i.e. paid for) only when needed
    - The sites extend their batch capacity overlaying commercial cloud resources
      - This is (slightly) less work for the experiments, but much less elastic
    - My preferred solution is to use clouds as clouds… and sites ensure the infrastructure exists… and they pay for what is used.
- Data storage and data movement are less affordable on clouds
  - HEP is mostly about fast access to very large datasets
  - Should storage remain owned by sites?
    - In this case adequate network bandwidth is necessary

Conclusions

- LHC Computing is probably a good example of Darwinian evolution
  - Several species started to be developed in the early 2000's
  - A lot of software was developed then perished
  - This is not the end of the evolution though
    - New species appeared recently and will appear in the future, others will disappear…
      - For example Rucio in ATLAS for Data Management
      - VAC / Vcycle pilot factories
- The survivals were essentially those that fitted the requirements
  - Top-down approach is definitely not what should be done!

- Nobody really knows what will be the trends in HEP Computing in the next decade, but the landscape will for sure change!

- It is absolutely necessary that experiments, sites and funding agencies are agile in following the trends
  - We are still largely using computing infrastructures like 20 years ago…