25th August 2017

Version 1.0

Facilities, Distributed Computing

Preface

This document represents input from the HEP community. **It is not a description of the computing models or facilities for HL-LHC;** rather it is a gathering of ideas for directions that should be explored in order to address the concerns for the future. As such there are many suggestions of how things could be improved, not all consistent with each other, and certainly not intended as the final answer. It should be taken as a guide for where the community needs to invest effort in order to understand what directions computing models and facilities could take in future, and to understand how useful, acceptable, and cost-effective they may be. Above all it should be seen as ideas for further discussion and exploration over the next few years.

Introduction

The LHC science program has utilized a globally federated computing infrastructure for the last 10 years to enable its ~10k scientists to publish more than 1000 physics publications in peer reviewed journals. It has also been used by other HEP applications. At present, this infrastructure comprises ~750k cores, 400 PB disk space, 600 PB of archival storage, as well as high capacity networks to connect all of these.

The basic operational model is that each of the four experiments produces primary data for their collaborations as a centrally organized process, and provides interfaces for the individual scientists and small groups to analyze this data to derive the results that are used in the peer reviewed science publications. Taking 2016 as a reference year in Run 2, the community processed roughly 10 Trillion collision events, often requiring multiple runs across parts of the primary data.

The computing hardware costs for the LHC in 2017 are dominated by the cost of disk storage, closely followed by CPU. The global investment into archival tape storage is modest in comparison. Networking costs are not well known, since for the most part they are not explicitly costed in the LHC budget. Naïve projections from current practice to the HL-LHC data volumes, taking into account Moore’s law cost reductions of 10-20% per year, predict that computing hardware needs exceed a flat hardware budget scenario by a factor 10-25. In response to this staggering problem of “affordability” for the HL-LHC computing, we are rethinking the overall LHC computing models.

The presently understood needs for HL-LHC resources are described in [***campana***].

A computing model as we have it today has a variety of cost and functionality trade-offs. The purpose of this document is to illuminate some of these trade-offs, and define possible R&D and prototyping efforts that would validate new models, and their technologies, in order to ultimately achieve a computing infrastructure for the HL-LHC that is as dramatically successful as what we have today while at the same time fitting into a constrained budget scenario.

The Cost Problem

As argued above, the major problem of computing for the HL-LHC is that of cost. In this section we attempt a simplified explanation of this cost problem. All experiments have detailed cost models that are updated over time in order to respond to their evolving computing models, in addition to simple adjustments to the model parameters to reflect reality. The primary objective of these models is to periodically estimate the computing requests of the experiments to the funding agencies. Despite their differences from experiment to experiment, and their complexity, they have a few ingredients in common that ultimately drive the cost.

Because of the diversities in the respective physics programs, the ALICE and LHCb experiments foresee a major change of data taking conditions already at the time scale of Run 3. For HL-LHC such conditions are not planned to change considerably and the computing resource needs will result considerably lower than what is foreseen for ATLAS and CMS. Therefore, for the remainder of this chapter we will focus on the latter two experiments.

The HL-LHC program foresees proton-proton collisions at the centre of mass energy of 14 TeV with an instantaneous luminosity of at least $5*10^{34}$ cm^{-2} s^{-1}. At Run 2 collisions occur at 13 TeV centre of mass energy and at most $2*10^{34}$ cm^{-2} s^{-1} luminosity. The difference in energy has little effect on the computing parameters, but the increase in instantaneous luminosity is such that at HL-LHC the trigger rate would need to increase by a factor 5 to 10 to be able to retain all interesting events (for example events with high p_T leptons). The event complexity would also increase because of the high number of collisions, and therefore primary vertices, per bunch crossing. At HL-LHC an average event pileup of 200 is foreseen, while in Run 2 this is not expected to exceed 60. The number of LHC live seconds in one year is expected to be similar to what we experience now i.e. 7.8M seconds in one year corresponding to 56% LHC efficiency over 160 running days.

The machine parameters are one of the main (but not the only) drivers of the HL-LHC computing costs. The main points to consider are:

- The total number of events recorded with the trigger system (i.e. average trigger rate times LHC lifetime for the year). For the HL-LHC the average number of collected events is planned to increase by a factor 5 to 10. To zeroth approximation, computing costs increase linearly with the total number of events triggered per year.

- The “average” event size on disk. Disk space in the current model is occupied mostly by “analysis formats” such as AODs and their derived formats (Mini-AODs for CMS, DAODs for ATLAS). The AOD size is mainly driven by information about particle tracks and clusters and therefore scales linearly with the number of particles in the event. Because of the increase in pile up of a factor 4 or more at HL-LHC, the AOD size is of the order of 1 MB/event for data and 20% more for Monte-Carlo (because of the truth information).

  On the other hand, a highly refined format that stores only information on objects from the hard scatter process would not grow in size as a function of pileup. Today such a format is being discussed in CMS at a size of <10 kB/event. If such a format can be realized it would dramatically change the total disk space needs at the HL-LHC, for most analysis work.

- The "average" event size on tape. Tape is currently at least a factor 4 cheaper than disk, so as mentioned in the introduction it is not considered one of the main cost drivers. However, our use of tape is not for archive purposes only, as we process roughly once per year data from tape. The tape cost for HL-LHC could change considerably if the market moves in a direction not compatible with our current tape usage pattern. In addition to the archive AOD formats, discussed above, LHC experiments rely on tape for the custody of the RAW data. The RAW event size grows linearly with the occupancy of the detector which again has a linear dependence on the pileup. Therefore we estimate an average RAW event size of 5 MB at HL-LHC to be compared with 1 MB per event during Run 2.

- The "average" CPU time to process an event. The experiments typically distinguish many such event processing cost numbers. There is at least one each for simulation, reconstruction, and analysis, though the full cost models are more complicated today. Reconstructing an event at HL-LHC conditions (pileup 200) with the Run 2 software through the current detector layouts requires 50 times more time with respect to an event at Run 2 conditions. Software re-engineering and tuning, combined with a different layout of ATLAS and CMS detectors in HL-LHC should diminish the increase to a factor 5 to 10, which is still considerable. The trigger emulation run in the Monte Carlo production chain will be affected by the same rough increase as it relies on the offline software. Detector digitisation foresees a linear increase of CPU need with pileup and therefore becomes negligible with respect to reconstruction, which has a more exponential growth. The time for detector simulation will not change between Run 2 and Run 4 as the pileup events are mixed with a signal event after detector simulation. Both experiments thus predict that reconstruction costs exceed simulation costs per event at the HL-LHC (by a large factor in CMS, less for ATLAS). It is to be noted that while reconstruction times in CMS and ATLAS are very similar, detector simulation in ATLAS is very CPU intensive and requires 10 times more than in CMS due to the different detector geometry and materials, particularly in the calorimeters[1]. Both experiments are evaluating changes in detector design and algorithms to control this growth in reconstruction time per event. Any R&D that achieves a significant reduction in this growth factor without affecting the physics program of the LHC has the potential of significantly reducing the HL-LHC cost problem.

- Related to both of the two previous cost issues is the analysis model. If an analysis model was viable that made use of a reduced event size and increased the typical analysis event processing rate for ATLAS and CMS from O(1-10) Hz to O(1-10) kHz then this would have a significant impact on the cost problem. It's maybe worthwhile pointing out that a simultaneous decrease in event size on disk and increase in processing rate could lead to facilitating I/O requirements that are similar to today while at the same time significantly reducing the cost problem.

[1] Refer to related Detector Simulation white paper.

- The computing resources needed for the event generators is becoming significant. Until now this has been a trivial CPU cost that has been ignored, however, with the need to reduce the systematic uncertainties due to theory, the generators are now introducing higher-order terms (NNLO) into the calculations, and these are causing the generators to be a significant consumer of CPU cycles, that must also be taken into account.
- The highly distributed nature of the LHC computing model today provides an opportunity for cost reductions due to aggregation. The sources of these potential reductions are due to less replication of data on disk, and consolidation of human resources required for operations of the globally distributed system at all levels. The risk posed by any aggregation is the loss of intellectual leadership at all levels, agency commitments at the various national agencies, as well as in kind contributions of power, space, cooling, networking, and human resources for operations. Ideally, computing model modifications would create opportunities for distributed ownership and leadership while at the same time creating cost savings through aggregations of services that drive the data replica counts, and thus drive costs inherent to the global distribution of resources.

Finally, it should be pointed out that cost is not the only problem the HL-LHC computing infrastructure development is facing. There are also challenges and opportunities due to technology changes. Among them are more heterogeneity in many dimensions, from computer architecture to the business models of commercial providers and HPC regional centres to various possible advancements in storage and networking technologies. The remainder of this document touches on aspects of all of these, trying to identify R&D and prototyping efforts that could have a significant impact on HL-LHC computing.

Current computing models

The CERN LHC experiments have evolved to have generally similar computing models at the conceptual level. While there are differences in terminology across experiments, the activities are much the same: Monte Carlo simulated-event generation (responsible for a large fraction of CPU usage), event reprocessing, group-level analyses, user analysis jobs. Data distribution has evolved from a hierarchical, Tier-based model where data products were explicitly allocated to sites, to a more demand-based model of distribution to support observed requests, in some cases backed by an automatic data federation system.

It must be remembered that the original computing models were defined before there was any experience in operating a globally distributed system, and the designs had to make allowance for all eventualities (e.g. poor networking, infrastructure failures, redundancy, etc.). In addition, there was no clear “correct” design of the experiments’ services and software. This is why there were very different solutions from all 4 experiments initially. Over time, the best solutions became apparent, and were implemented independently in each experiment. The situation today is far better understood in terms of the core functionality that HEP needs, as well as the fact that industry is also running global computing infrastructures, with a lot of expertise and experience now. There is now a clear opportunity to benefit from that understanding, and to re-engineer common systems that support all of the LHC experiments, and hopefully other HEP collaborations.

The original hierarchical Tier system of WLCG has been well described elsewhere [***monarc***]. The definition of the “Tiers” has also evolved; in general, there are now Tier-1s and “everything else”. The distinction for a Tier-1 is largely driven by provisioning of long-term (archival) storage which requires experience and investment, the expectation of a professional team that monitors, operates and supports the facility on a 24/7 basis, and a long-term commitment to the project. This does not mean that other sites might not be professionally operated, etc, however a Tier-1 possesses all these attributes.

The bulk of the computing and storage resources used by the LHC experiments resides behind grid-type gatekeeper interfaces. As of 2017, the scale of the officially-recognized resources (Tier 0, Tier 1, Tier 2) is about 5200 kHS06 (~500,000 modern cores[2]) of computing, 400 petabytes of disk, and 600 petabytes of tape. The Tier-1s are located in North America, Europe, and Asia; Tier-2s are found on all continents with the exception of Antarctica. Significant additional computing is opportunistically acquired from additional non-pledged resources in several Tier 1 and Tier 2s, as well as cycles on supercomputers, available through limited discretionary allocations, or as backfill to larger MPI job-sets. One significant regional difference is that most North American sites predominantly serve (or are sometimes even dedicated to) a single LHC experiment, while in Europe most of the facilities support multiple experiments, often alongside other science groups from completely different fields.

The gatekeeper interfaces are several: ARC, CREAM, and HTCondor for computing, and SRM, http, gridftp, and xrootd for storage. On the other hand, the underlying computing platform is generally quite homogenous: almost all resources are running some variant of (at this writing) RedHat Enterprise Linux.

The diversity of gatekeeper type interfaces is traceable to development opportunities funded by the large grid-computing projects in the previous decade. While they at some level do exactly what we need (get a job running on a remote system, retrieve a file from a remote system, etc), they are in general overly complex for the task at hand, while on the other hand not providing all the functionality we need. For example, accounting of storage usage is still done manually, and memory allocation requests are not uniformly handled across the computing interfaces.

Need for evolution

Despite these well-known deficiencies, the WLCG grid of today has served LHC well, and enabled the huge physics output of the experiments. The computing system is a fundamental cornerstone of producing physics results and has been recognized as such.

There are a number of factors which will drive an evolution of the current implementation of the distributed computing infrastructure, as well as some barriers to scaling the system to the perceived needs for the HL-LHC era.

As understanding, and reduction in complexity, of the running system has developed, some of the choices made in the past may no longer be relevant. In many cases the existing

[2] If purchased today there would be some 500,000 cores, in reality there are more as many are much older and less performant.

implementations of several services will not scale to match the needs foreseen in the coming decade. Thus there is both a need to rework some of these, but also an opportunity to further reduce complexity, support needs, and to consolidate differing solutions. It is important to capitalise on this opportunity.

In addition, the future infrastructure needs to be capable of easily adapting to changing technologies, especially to disruptive new ideas.

The greatest enemy of the current system is complexity. As already mentioned, at the grid-middleware layer, there are multiple external-facing interfaces to both the computing and storage resources. These must be interfaced with a second level of site-internal resources management systems, for example for computing we have SLURM, HTCondor, LSF, Torque, etc. Due to the gaps in functionality provided by the gatekeeper interfaces, experiments must sometimes deal with these site-internal details. Above this infrastructure layer, each experiment has its own specific data-management and job-management systems. At each layer, the software must deal with many possible permutations of possibilities.

There is a related pressure on computing manpower. Experiments generally request a “computing person” specific to the experiment to be located at each site, who can deal with the experiment-specific debugging chain of problems related to that site; a single person cannot be expected to learn the ins and outs of several different experiment frameworks, nor can a single experiment person be expected to master the permutations encountered at several different computing sites.

The development of edge services, and an associated edge services orchestration platform, is seen as a potential technology to facilitate automation (e.g. deployment and updates) and operations (service lifecycle management) for multi-site platforms. This reduces the experiment-specific computing manpower required at the site while increasing the frequency of release updates, allowing for a more DevOps model for distributed systems.

Scalability limitations

At the time of writing (Summer 2017), the scale of resources thought to be required for HL-LHC is approximately 60x larger in compute than that available in 2017, and around a factor 10 or more in storage. The working assumption is that there is a factor of 10 between what will be required and what is expected to be delivered by technology evolution with constant budgets. That is the key challenge.

It is impossible to reliably predict problems for scale increases of more than a factor of 5 or so, given the complexity of the current system, however we can enumerate some of the ones we already know about. The most important one is associated with storage and management of the data. Firstly, an HL-LHC version of the current system would require a factor 10 expansion in disk capacity. Secondly, effective use of this storage relies on rapid access, while the effective bandwidth per TB capacity is decreasing for current disk architectures. The do-nothing solution is to only partially use the storage capacity, which raises the effective bandwidth per TB and also raises the disk expansion factor above a linear increase. The community considers it unlikely that funding agencies will be able to finance such an expansion in storage scope.

The other main limitation is manpower. An HL-LHC version of the current computing model plus infrastructure ecosystem wouldn't require a linear scaling of the manpower, but it's unlikely that the current (operations) manpower level would be sufficient. This contrasts unfavorably with present difficulties in sustaining the current level of computing manpower. A more complete discussion of scaling problems

Funding situation

As a general guideline, advice from the funding agencies has indicated that a scenario of constant (flat) funding should be used for planning computing hardware needs for the foreseeable future. **It is important that we agree that a flat budget scenario means flat across all years whether LHC running or shutdown (or equivalent integrated over time)**. Otherwise the technology advances that we rely on will not be affordable.

At CERN, the Tier 0 funding is part of the medium-term plan (5 years) for which the expectation is constant budget, in line with other countries. It is, however, possible to re-profile expenditure within that envelope, so there is some flexibility, which makes managing resources between running and shutdown years easier.

National Funding (Tier-1 and Tier-2)

The worldwide ecosystem of nationally-funded LHC Tier-1s and Tier-2s is well funded at the required levels, generally speaking. At the experiment level, this is not always the case; due to differences in participating countries and (sometimes even within the same country) different funding agencies, some experiments have acquired significant amounts of "non-pledged" resources while others have difficulty acquiring funded computing at the required level.

While successful, the situation is certainly not comfortable, as in almost all cases, the funding is provided in slices of one or two years, often associated with a specific project. Even at the scale of the current LHC, seen logically, computing resources should be a recurring budget item in the science budget for each country providing a significant resource contribution. As far as we know, this is nowhere the case. Furthermore, none of the experiments has a model for crediting computing contributions, e.g. as equivalent M&O contributions; in some countries, the financial value of the computing contributions already exceeds that of M&O responsibilities. A significant increase in the scale of computing resources due to HL-LHC will likely require sustainable models for both the national funding and the experiment credit.

Computing Operations Funding

Here as well, there is no uniform model, and information is scarce, as budget discussions at the WLCG level focus almost entirely on funding the resources themselves. A couple of general observations can be made:

- Except in those cases where funding for computing operations manpower is acquired as an integral part of the resource funding, such manpower comes at the cost of other collaboration activities, such as analysing data and doing physics. In this respect, evolution of the system to require less experiment operations manpower is a desirable goal.

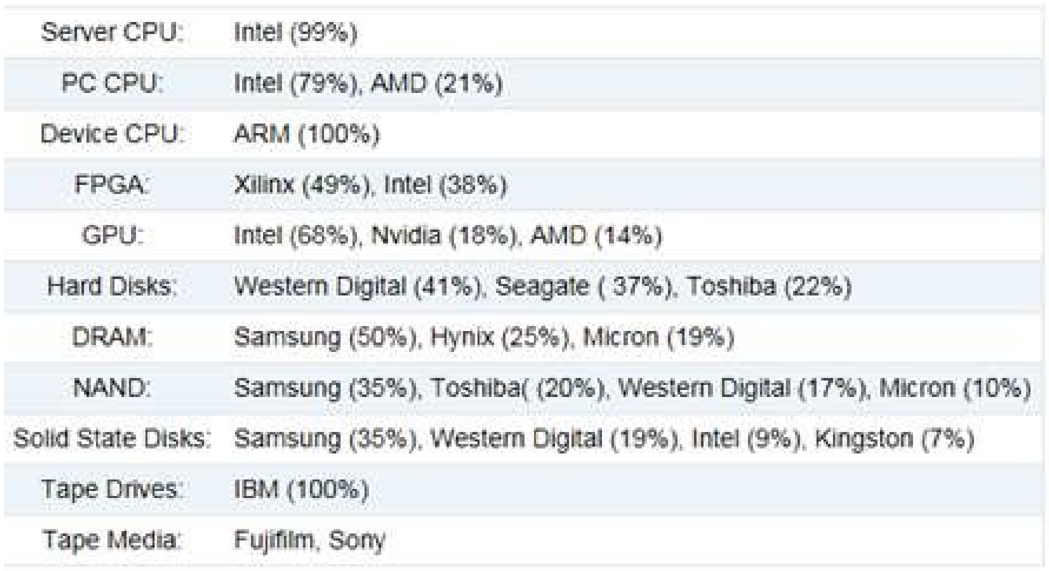
- Dedicated facilities (experiment-specific or LHC-specific) are generally completely funded by high-energy-physics funds; in these cases, the facility operations manpower is also a cost to the experiment collaboration. Housing of LHC computing resources within a larger, national, generic scientific computing facility generally removes some of this cost, as the LHC facility operation becomes an (albeit sizeable) increment upon an existing facility, and hence may rely more on national, non-HEP funding.

Evolution of the computing landscape

Technology and Markets

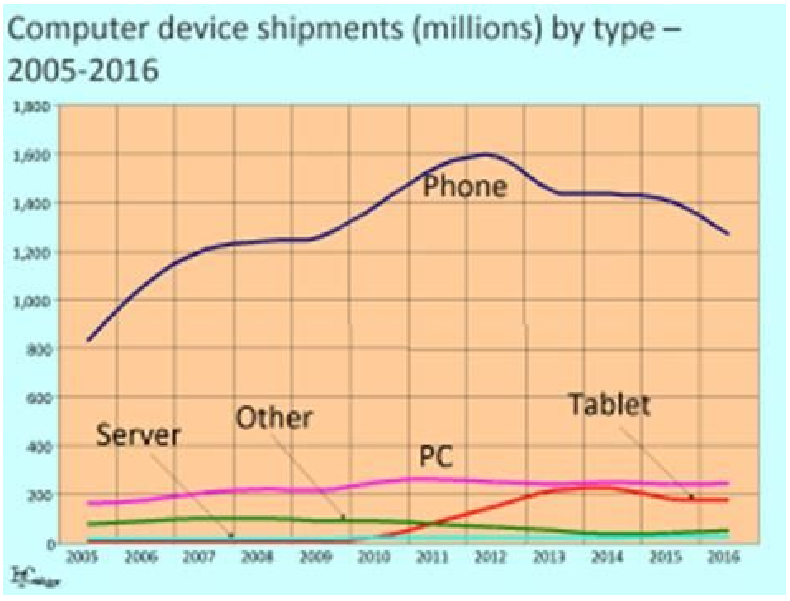
Industry developments

The current state of the computing industry is characterized by aggregation and consolidation of companies in the various sectors. Only very few companies are dominating the markets.



Server CPU:	Intel (99%)
PC CPU:	Intel (79%), AMD (21%)
Device CPU:	ARM (100%)
FPGA:	Xilinx (49%), Intel (38%)
GPU:	Intel (68%), Nvidia (18%), AMD (14%)
Hard Disks:	Western Digital (41%), Seagate (37%), Toshiba (22%)
DRAM:	Samsung (50%), Hynix (25%), Micron (19%)
NAND:	Samsung (35%), Toshiba((20%), Western Digital (17%), Micron (10%)
Solid State Disks:	Samsung (35%), Western Digital (19%), Intel (9%), Kingston (7%)
Tape Drives:	IBM (100%)
Tape Media:	Fujifilm, Sony

All device markets show very low increases or negative growth rates with clear signs of saturation.



http://www.itcandor.com/storage-2016/

Processors

The processor market is dominated by Intel in the PC and server area and ARM in the general device market (phones, tablets). So far, all attempts to break the Intel monopoly in the server market with PowerPC, ARM or AMD have failed. More competition is expected towards the end of 2017 with the introduction of new models from AMD and the 3rd generation of specific ARM processors (Qualcomm, Cavium, etc.).

The current fabrication of microprocessors is using the so called "14nm node" which corresponds to structure sizes of 50-70nm on the chips. The next generation of 10 nm nodes is already tested and will go into full production next year, with processors based in 7nm technology planned for 2019/20. IBM has shown prototypes of 5nm node manufacturing. The speed of the manufacturing improvements has slowed down considerably during the last years. The market leader Intel was using a Tick-Tock model in a 2-year interval, one year for a decrease of the structure sizes on the chip and one year for improvement of the microprocessor architecture. This has moved to one-year structure size improvement followed by three years of architecture changes.

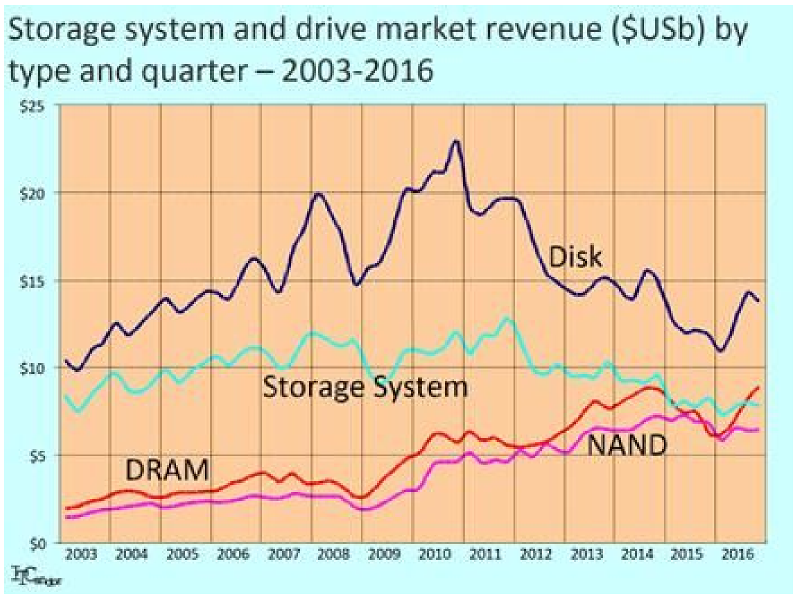
There are currently only four companies (Intel, Samsung, Globalfoundries, TSMC) at this level of structure sizes (<=10 nm) which can afford the very large (~10 B$) capital investments for new fabrication facilities.

The server market (10 million units and 55 B$ revenue per year) shows clear sign of saturation with very small or no growth rates.

We probably can still expect a 20% price-performance improvement for CPU servers over the next years.

Disk storage

Falling revenues and unit sales currently characterize the traditional HDD storage market due to the Desktop PC market declining now since 5 years, increasing usage of SSDs in Notebooks and a strong competition of SSDs in the high-end enterprise disk area.



http://www.itcandor.com/storage-2016/

The technology change from PMR (Perpendicular Magnetic Recording) to HAMR (Heat Assisted Magnetic Recording) with much higher bit densities has been delayed and now first such HDDs are anticipated for 2018. In the meantime, two intermediate technologies are used to increase bit densities at the 20% level (SMR and TMR).

For the next years, one can still expect a cost improvement of about 20% for Hard Disk storage.

The cost evolution of SSDs is also improving rapidly. However, for large capacity disks the standard HDD will still have a cost advantage of a factor 10 for at least the next 5 years.

Tape storage

In 2017 the cost for LTO tape media has reached 0.01 Euro/GB and shows an improvement rate of about 20% per year. The latest report from the LTO consortium shows a continuously decreasing amount of tape media sold while the amount of cumulative space is still increasing (10 EB per quarter compared to 150 EB for HDD). After Oracle stopped the development of Enterprise tapes in the beginning of 2017, there is only IBM left manufacturing tape drive heads for LTO and IBM Enterprise tape drives. The yearly production of heads is currently about 230000 compared to 450000 in 2014. The only two companies (Sony and Fujifilm) manufacturing LTO tape media are entangled in a patent ‘war’ since 2016. The future development in the tape area needs to be watched closely. A move from tape to hard disk storage would increase the costs by at least a factor 3.

A more detailed report about the technology and market evolution of storage can be found here: https://twiki.cern.ch/twiki/bin/view/Main/TechMarketDocuments

Summary of hardware evolution

Moore's Law and Kryder's law have slowed down. The Device and Server markets show signs of saturation. Technology evolution is also slowing down due to the increasing complexity and large-scale investments needed.

The current expectation for price-performance improvements over the next years are 20% for CPU server, 25% for disk storage and 20% for tape storage.

More details and all reference can be found here: https://twiki.cern.ch/twiki/bin/view/Main/TechMarketPerf

Commercial resources & Impact of Internet services

Another big change that we have seen since the inception of WLCG is the emergence of large Internet companies, not just providing commercial services and cloud computing, but also developing globally distributed computing infrastructures, and the tools and expertise to manage and operate those. This was missing at the outset of the LHC computing projects, and HEP had to develop from the ground up the tools to run computing infrastructure at unprecedented scale. Over the last 5 years or so, we have been able to gradually introduce many of those open source tools into our operations, and so benefit from all of the accumulated experience. Such tools also enjoy a huge support effort, far larger than HEP can imagine doing alone.

These companies have also normalised the idea of having computing resources "somewhere else", and available over the network. This, while one of the dreams of the grid, has only actually been truly realised with the development of various necessary technologies (e.g. virtualization/containers), and has pushed the evolution of the commercial internet to support such remote services. HEP has been able to utilise the technologies to good effect in some cases, but today HEP still sees having data sets and processing capacity close to the physics users as very important.

Such technologies will change the way HEP provisions and accesses computing and storage resources in several different ways (and in many cases this is already happening):

- Use of the tool sets to help manage our resources, with tools that have wide support and development efforts; examples include the devops tool sets, provisioning tools such as OpenStack, and many others. This change has already happened, and will continue. The need to develop these tools ourselves (alone) has gone, and we are part of the broader communities which will help assure sustainability;
- Change of mindset from having data locally, to working with remote data. This could take many forms, and has already been demonstrated in WLCG with various data federations. In the future this may become more explicit, by ensuring that bulk datasets are stored in relatively few places, and served to the global community rather than replicated many, many times. This idea is developed further in this and related whitepapers. This is the model of data "in the cloud" that is the heart of many of the commercial cloud providers.

- Direct procurement of commercial resources from cloud (or other) commercial providers. This has already been demonstrated to be practical, and scalable to order several hundred thousand cores. How the uptake of this develops will really be driven by the cost of such resources. Different cost models exist, from long term "rental" to "spot-market" style provisioning. Different modes may be applicable in different scenarios. In particular, for provisioning peak loads, this kind of commercial resource through a spot-market mechanism may well be very soon cost effective.
- In addition to the previous point, which essentially imagines a large site (or group of sites) procuring such resources, there may well soon be the move of for example national infrastructures which could potentially outsource all of their academic computing to commercial clouds, and providing credits to researchers to get access. This may become cost effective, especially for usage patterns that have sporadic needs for computing, or need services to be provided, where the investment in a computer centre at institutions may not make sense, especially when operating costs (people) are accounted for. If a country decides to provide the LHC contribution in this way we must be agile enough to utilise it.

HPC & Specialised Resources

It is becoming more and more common that HEP is being offered cycles on large HPC machines. This has been the case in the USA for several years, but is now seen in other countries too. While most of our applications are not massively parallel to make good use of the types of hardware often found on those machines, nevertheless there is pressure on HEP to make use of spare cycles, through backfilling of unused cores, or by actually making explicit requests for time on the machines. Several countries are considering consolidating (large scale) scientific computing onto machines of that type. This would imply that the contribution to LHC would come through that provision. A lot of work has gone into making many HEP applications run on such architectures, but each of the machines are idiosyncratic and require specific setup. HEP must make sure that we are easily able to adapt to and use such resources when available, both from the point of view of the infrastructure, but also through making our applications efficient on such hardware.

Generalising this point to other specialised types of machine, we may see that rather than the ubiquitous x86-style architectures we have made great use of in recent years, we will find more and varied specialised hardware. HPC machines are one example, but GPU clusters, clusters integrating FPGA, or other novel hardware types may be made available. In this case we will need to understand whether the investment in porting software to those machines makes sense, which will depend on many factors - skills, ease of porting, availability of the resource and how long it may last. We may see different such specialised machines more suitable for specific tasks, so we could see the evolving of centres of expertise or competence for specific workloads and hardware.

Other Influences

Other factors will also influence the resources that we need for HL-LHC computing and the tools that are used. Already we see in the broader scientific domain (as well as HEP) a huge take up of so-called "big data" tools, a massive interest in Machine Learning/AI techniques and so on. These new analysis models will have a potentially huge influence on the types of facilities we will need (as well as data management tools). For example, the training needs of the ML/AI tools could well be most effective on certain GPU machines. How to balance these various requirements for specific rather than general purpose machines will be important to understand, and must be part of the cost analysis.

Networks

In particular how academic networking develops will be a key factor in defining the eventual computing models. Today, the LHC traffic is essentially the main player on most academic research networks. This is likely to change over the coming decade, with other large data volume scientific endeavours coming on line. HEP will have to share the bandwidth, and balance between the pressure to help develop the networks for our needs against the potential costs if bandwidth becomes contentious. Here we may distinguish between a large bandwidth between key data storage locations, and the needs of the remote locations where the physicists work. Today's model requires "large-enough" bandwidth to all Tier 2s in order to make their CPU capacity effective. Having a more data-centric model (data-cloud) may be more cost effective.

In addition, for a given cost, a generation change in capacity (currently 100G to 400G) takes ~8 years. This translates to roughly 22% per year, much less than the solid >50% per year growth in traffic in observed in the main academic network backbones in Europe and the USA. Currently there is no technology breakthrough on the horizon, and indeed 800G already produces an energy density challenge in the optical interfaces and switching chips. The message is that the network situation will change starting already in the next few years, perhaps during Run 3. The computing models must take this into account in order to be efficient.

Funding

Finally, but very importantly, the way in which funding available for LHC computing may be spent will have a great influence on the models. The original model of national funding spent locally in building local resources, may no longer be appropriate. Several countries have started discussing how to consolidate their scientific computing infrastructures to be more cost efficient, including the use of commercial resources, and large scale (HPC) centres. In Europe we have demonstrated the possibility of doing joint procurements of commercial resources across funding agencies, and across disciplines (HNSciCloud project). The willingness of funding agencies to consider new mechanisms of funding computing will be a key for achieving an overall cost optimisation of the needs of HL-LHC.

Strategy towards HL-LHC

The strategy for providing a computing environment adapted to the needs of the HL-LHC era, must be one of continual evolution of the running system. Research and development efforts must continually feed into that evolution, bringing immediate benefits. Replacement of major software components is possible and has already been done by all of the experiments.

In order to drive that evolution, we must have a process of continually monitoring the system in order to know where next to invest effort. Today one of the major costs is related to the amount of data replication and distribution that is needed.

The model must be able to continuously adapt to changing hardware, technologies, funding situations, and the overall ecosystem. This includes needed agility towards the use of all possible and available resources, in-house, commercial (leased, cloud, etc), HPC, volunteer, various forms of opportunistic resources, and so-on.

As discussed further in this document, the need to consolidate the major software components into common services serving all experiments will be vital. The opportunity comes from the good understanding of the basic requirements which are really common, and a general recognition that such commonalities are essential to reduce the support and operational costs, as well as to be able to strongly focus on understanding and improving performance and functionality - for the benefit of the entire community.

It is clear that we must develop the infrastructure so that it can manage both large scale data storage and serving, and make use of very large and diverse as well as small scale and opportunistic resources.

The infrastructure should enable new analysis models other than the traditional sequential “event-loop”. Today these are focussed on big-data style tools, machine learning, etc., but we must maintain a capability to adapt and incorporate new technologies as they arise and are seen to be useful.

In order to achieve such an evolution, we must invest sufficient effort into R&D and building prototypes, which can be tested, and then adapted into the running system.

Today there is a huge bottleneck to any significant change, particularly changes related to code execution, underlying architectures, etc. That is the need for physics validation of the results. This is a hugely time-consuming and manual task. In order to achieve the needed adaptability, performance, and evolution that problem must be addressed, so that the validation can be far more automated.

Thus there will be no final computing model for HL-LHC, there will be a strategy of analysis, evolution, prototyping, and measurement that will continue for the lifetime of HL-LHC.

Cost Models and Cost Management

To understand a cost model it is important to understand that the running system that we have today must continually evolve into the computing system and computing models that are needed for HL-LHC. There will not be a single step-change. For this reason results of R&D work must feed into, and gradually improve the running system (in the way that it has during Run 1 and Run 2). However, that should be guided by cost arguments in the broadest sense.

Thus, the idea of a “cost model” for computing will likely take the form of an iterative process that builds understanding of where the main costs are and where the performance and functional constraints lie. The ultimate goal of having a cost model is to be able to understand where to invest effort in improvements or changes, and how to best optimise between investments in different functional elements of the overall system. This will be a continually evolving process, and needs to make use of understanding gained from the following key components:

- Studies of end-to-end and specific and functional performance measurements, deriving appropriate metrics;
- An understanding of how hardware functionality and cost will evolve over the coming years;
- An understanding of where the main constraints in the running system are, and what would be gained from addressing those, or changing the system;
- Consideration of costs that are today ignored, but that affect the overall cost to the community: networking, operational effort, maintainability of the service or code, and long-term sustainability;
- Understanding the balance between investments in hardware or services and investment in improving application software performance.

These factors must all be measured and monitored with agreed metrics and used to analyse how to best make investments. In this way we will gain understanding of the costs (hardware, operational, functional compromise), and hopefully be able to better optimise the efficiency and functionality of the service. This is something that has been missing so far. With sufficiently detailed metrics inherent differences between sites can be taken into account, as cost optimisations may not be the same everywhere.

Metrics

It is important to look again at the metrics that we use to analyse the performance and effectiveness of the system. Today we are hung-up on reporting CPU use and CPU efficiency, partly because we think we understand what they mean. However, in order to do a real optimisation of cost, we need to be able to make an effective judgement of where to invest funding. We must build a set of useful and understandable metrics that allow us to make objective judgements of efficiency and performance.

The main areas in which it is important to be able to understand the costs arise from the main elements of the computing model problems which are well-defined steps:

- Data Collection: the cost to record and store data
- Data processing: the cost of the initial reconstruction and re-processing data paths required to write the formats used for analysis
- Simulation: the cost of producing both full and fast simulation
- Analysis data reduction: the cost of data access and to select and reformat the data samples for more detailed analysis

As understanding is gained, the model can be refined and improved.

Facilities and Computing Models

As discussed above, the facilities used for the collection, simulation and analysis of current LHC Run 2 collisions data is based on a rigid organisation of sites, still named after the MONARC hierarchy.

The Tier-0 facilities, unique since they are close to the experiments, are mainly used for fast calibration and prompt processing, providing the physicists with data to be analyzed a few days after data taking; they also serve as custodial tape storage for the incoming detector raw data. Its Service Level Agreement (SLA) is 24x7 during data taking activities.

Tier 1s store the second custodial copy of raw data, and are used for intensive data processing activities like Monte Carlo processing and data reprocessing. They also offer 24x7 support levels.

Tier-2s are closer to the end analysis users, and serve as analysis facilities and Monte Carlo processing centres; their SLA is 9x5, compatible with office hours.

In reality, the boundaries between the hierarchy levels have substantially blurred since the start of LHC Run 1. Currently, every facility is able to run most (if not all) the experiment workflows, while distinctions in SLA and funding scheme in most cases is still important.

The capability to run any workflow at any site, eventually reading data via geographical WAN paths, helps to increase the CPU utilization, and at least partially decouples workflow execution from data location; we have experienced a growth in adoption of this paradigm over recent years, and we can expect this to become more and more viable as the general / guaranteed network connections improve.

The current facilities model, apart from initiatives mostly at the scale of tests, expect static resources, where sites take time to be commissioned, and are expected to be long lived. In the next decade, the expected funding situation anticipates the need for flexible resource provisioning, including short-lived resources and time-limited opportunities and collaborations. We can thus expect the need to move away from a rigid facility deployment, to a scenario with resources eventually joining and leaving the pool available for an LHC experiment at a fast rate.

Some issues of the current computing/facility model

The initial model had essentially homogeneous capabilities at most centres, differentiated mainly by scale. Actual use in operation showed that this is expensive to operate, and that efficiencies could be gained by focussing some workloads on larger sites with more storage. In addition the strict MONARC hierarchy was unnecessarily restrictive and relaxing that to a mesh model early in Run 1 reduced network traffic and improved efficiency.

The overall cost and performance is very hard to estimate or model, network requirements are not well estimated, as the model has been in flux and not synchronized with WAN upgrades (although it does drive the need for upgrades). Similarly, local I/O performance was not always considered seriously in contrast with typical HPC centres which are MPI-centric and so the I/O mesh is fundamental. There are many reasons for this, especially the desire to make the facility general enough to support a wide variety of workloads.

The nature of the dominant workloads of existing compute centres has meant that the centres have been quite inflexible to the needs of modern analysis tools - “big data” techniques for example. This adaptability needs to be included for the future as such analyses will be key to the centres evolving to follow broader trends.

As noted earlier, the HEP-specific nature of a lot of the tools has made it operationally higher cost, although this is already changing.

Future models are likely to need to integrate many more resources that HEP does not control; thus they must be flexible towards evolving the processing framework and data/storage models, as well as considering how the experiments organize their data versus how the facilities must organize (and provide access to) storage.

Facilities and models for the future

Given the above considerations, the infrastructure must guarantee a solid and performant access to the data, allowing it to be accessed and processed from resources that are not known a priori.

Some of the trends we have seen already with the LHC system and in the broader world include:

- Federating data, which implies the late binding of data to a CPU, and makes use of direct read access, or copy, over the network, and potentially caching data locally.

  This is a recent capability over pre-placed datasets that was not part of the original model, although it is network intensive and client sensitive, in some situations it improves the efficiency of the overall workflow (e.g. if some piece of data is missing locally). Clearly it has limitations, especially the perceived inefficiency of a CPU waiting for the data. This may be an overall optimisation, but the metric to demonstrate that is missing.
- A model of concentrating data at a few big centres (so-called “Data-Lake”- see e.g. [**HEP-CWP-009**, **HEP-CWP-004**]), following large cloud provider models. In this case smaller centres or dynamic-use centres access data more dynamically (or remotely, or using explicit temporary caches). It potentially offers new methods for skims and queries, and potential reduction in data management costs. It would be driven by or coupled to new models for organizing data, such as the use of Objects over files; the coupling to new storage infrastructure paradigms and their analysis frameworks (e.g. hdfs, object stores, in various implementations); and different models for coupling CPU to data (e.g. an effective use of a storage hierarchy).
- Increasing heterogeneity of services: finding centres that focus on providing specific services well with local expertise, rather than raw resource capacity. An example might be providing Machine-Learning analytics as a service.
- Increasing alignment with industry trends, following trends in data centre provisioning. Use of concepts such as LHC "Availability Zones" would help evolve from rigid hierarchical models to coarse regions determined primarily by power costs and WAN latency. Controlling costs would be helped by leveraging widely used facility provisioning tools to control costs and acquire effort (already today). Finding staff with experience on industry-wide tools is easier.
- Use of “Leadership” computing facilities (HPC/supercomputing) assumes that the existing trend of leveraging the investments of some funding agencies in large supercomputers will continue. We will need to understand how the cost is folded into our cost models.
- Use of commercial and opportunistic resources - for peak times or for specific lightweight tasks, organised in such a way to be able to use spot-market pricing,

low-cost or other opportunistic resources. Includes volunteer computing, commercial spot markets, shared backfill capacity, opportunistic resources.

- The evolution of "programmable" facilities. These are models which exploit advances in software defined networks and (multi-) data centre virtualization to realize additional capabilities, optimizations, and operational cost reductions. The facility would present cpu, gpu, storage and network resources with service interfaces which can be controlled and developed within the experiments. This would include things like the ability to deploy data resources (caches) in various places like regional NOCs to optimize data delivery, and with an emphasis on evolving edge service capabilities - "cyberinfrastructure as code". Examples include the extension of Google's "datacentre as computer" concepts to multiple centres (c.f. https://research.google.com/pubs/pub35290.html). The ability to create and destroy task-specific platforms on demand for particular campaigns might also be effective, by incorporating elastic capabilities to reach providers (hpc, cloud, commodity, campus) into configurations optimized by task function. This would give a flexibility to combine resources across funding sources, and to provide the ability to aggregate dedicated (project owned), agency allocations, commercial providers, institutional allocations (from a university research computing centre), and opportunistic resources.

We can distinguish between a number of prototype facilities we expect to see over the next ten years.

1. HEP-owned large data centres: defined as sites with long-term commitment to the Experiments, with secondary storage and good connection to primary storage (either integrated into the site or external), linked by high performance networks to the primary source of data and between them. Serving data to other centres is the primary role of these sites, either via the guaranteed links to sibling centres, or via less-guaranteed links to smaller centres. The centres would probably host sizeable CPU installations, even if this is not compulsory. Taken together, these centres would constitute the infrastructure guaranteeing solid data custody, fast ingestion of data produced either by the detector or as the output of processing, and data serving to all the CPU clients. Adequate guaranteed bandwidth, with appropriate QoS and traffic management will be necessary between these sites, (similar to today's LHCOPN).

   Ideally, from the point of view of the experiment, these centres would be seen logically as a single entity, with an intelligent service layer handling the data distribution driven by policies and QOS. Managed data movement would happen only within and between these sites' storage. This is what is referred to as "data lake" in other parts of this document. The "lake" is expected to be a complex addition of site storages, with storage services and capabilities deployed via a match-making between available resources and QoS for the ingested data.

2. HEP-owned small-to-medium sized data centres: they would be serving the experiment with CPU resources, and would be plugged to the large sites via performant network links, not necessarily guaranteed (can use the general academic network). Storage at theses sites would be present, but not managed by the experiments' data management systems, nor seen as a part of the data-lake. We can expect the storage at these sites to serve primarily 2 purposes:

   a. Serve as a cache at the site's boundary, in order to minimize network usage and decrease latency;
   b. Serve as local storage for analysis users private data. Public output of local processing, instead, would be ingested back to the data-lake for uniform access.

   The centres, being owned, are expected to be long-lived, and would deploy a sizeable fraction of the total CPU resources; again, the storage, if present, can be sizeable.

3. Sites with elastic resources. These can be:
   a. Commercial clouds provisioned by national funding (as discussed earlier) or by individuals through grants;
   b. Resources owned by Research Institutes not directly participating in an LHC experiment, but willing to offer CPU cycles at specific moments;
   c. HPC sites providing allocations to the collaborations;
   d. Other types of opportunistic resources.

   All these cases are in general not too different from the second-type data centres, but are generally expected to be short lived, and can appear/disappear at fast rate. Since they do not include any managed storage, a sudden loss of one such centre has only impact on the running jobs which can eventually be evicted.

As a general guideline for the computing at these facilities, it is imagined that that managed data only sits in the first type, and its complexity is masked by the existence of a single logical entity (the "data-lake"). All the sites in the second and third category are considered "Compute Sites", and are stateless at the granularity of single jobs.

The major ingredient in such a facility structure is the availability of adequately-performing network paths, guaranteed or not. Outside the data-lake, all accesses to managed data is expected to be via streaming, eventually mediated by site level caches.

The foregoing discussion sets the scale and the types of deployed sites, but not their specific resource types. Currently, all the resources used by LHC are very similar: x86_64 worker nodes, with 2-4 GB RAM per core, some 10-20 GB/core of local disk, and a compatible operating system (usually RHEL/CentOS derived) - as specified in LHC Experiments VO cards.

For LHC Run 4, we expect the need to provision resources of more diverse types and different operating systems; even if that would not be a necessity, we can expect such resource to be available at least on the opportunistic level in the form of specialized sites, HPC sites, or similar.

Such sites might include:

1. HPC sites, with hardware accelerators (GPU, KNL, Deep Learning specialized hardware) close to CPUs and/or fast interconnect between nodes;
2. Sites providing programmable DSP/FPGA close to CPUs;
3. Sites with special infrastructures like storage optimized for Databases and/or data mining workflow (Spark, Pig, Deep Learning);
4. Sites optimized for local data access (with SSDs, NVRAM, …).

In all cases, such sites need to be utilized optimally by the experiments. Apart from requirements on the experiments' software (which needs to be able to utilize heterogeneous architectures and resources), this poses requirements on the capability of Workload Management systems to be able to direct workflows to specific and particular resources.

Networking

Considerations

Networking, an area that was considered as a potential bottleneck at the time of the MONARC planning discussions in the late '90s, has become, at least for some *[reference CMS presentation at pre-GDB]* a limitless resource. From a purely technical standpoint *[see technology development section of CWP]* we see no reason why "the network", whether at the local- or wide-area level, should not remain an enabler for HEP computing. As far as internal site networking is concerned, therefore, adequate network capacity should continue to be affordable. Unfortunately, however, there are four non-technological issues that could adversely affect WAN transfers—the network capacity share allocated to HEP, (N)REN acceptable use policies, security, and applications treating the network as an infinite resource.

Network Capacity Share

Today, HEP networking demands are still, at least within the Research and Education domain, very significant. LHC-related traffic, for example, represents some 35% of the data carried by both ESNet and GEANT. However, as pointed out by some [see, for example (Tony Wildish, HEPIX LBL)], HEP data volumes are growing slowly compared to genomics datasets. Also, even if there will be significant on-site data reduction, attention is turning to the networking challenges being posed by other "big science" instruments such as the SKA.

Within a few years, therefore, HEP traffic may no longer see essentially limitless intra- and inter-national bandwidth but, instead, be constrained to operate within a bandwidth limit set either by financial limits or by some "inter-science" prioritisation. Here, it is useful to remember that the LHCONE infrastructure was not setup to facilitate LHC data transfers but rather because, in the era of 10 Gbps transatlantic circuits, network operators looked for a way to control their alarming growth and engineer potentially less-expensive means of delivering bandwidth the LHC. Fortunately, 100 Gbps circuits became available and HEP traffic volumes again became manageable but the point remains that LHCone remains a tool our network service providers could use to manage HEP bandwidth use.

Again, we are not suggesting that HEP bandwidth use will grow faster than technology can deliver increased capacity. We are suggesting, though, firstly that total research and education bandwidth demands saturate the capacity that can be afforded by the R&E community and, secondly, that *the share of this capacity allocated to HEP*—either on the basis of dedicated HEP funding or according to the relative priorities of different research areas—means that the network could become a bottleneck for inter-site transfers.

(N)REN Acceptable Use Policies

We expect commercial cloud service providers, primarily IaaS but also other flavours, to play an increased, and perhaps increasing, role in the future provision of HEP computing resources. Today, some NRENs have acceptable use policies that do not permit data transfers between their client R&E institutions and commercial companies---and amongst those that do permit such transfers, many require one end-point to be at an R&E institution, so forbidding, for example, transfers of scientific data between commercial providers via their network.

Unless such AUPs are relaxed, institutions will have to make an increased reliance on commercial network connections, an increased reliance that would likely come with increased costs. Complexity would also increase, most obviously in terms of the requirements for managing network connectivity at an individual site but also, for sites that provide capacity through a mix of local resources and “cloudbursting”, in the arrangements necessary to provide a consistent view of their resources to all users.

Security

One security related aspect follows directly from the point above: can commercial cloud resources be trusted? Today’s firewalls cannot handle the LHC data rates and so most HEP sites implement, in one way or another, the concept that has become known as “the Science DMZ”, locating computing and storage resources “outside” the network site boundary and limiting access to a set of trusted clients—specifically, for HEP, nodes connected to LHCone. To date, however, no mechanism has been implemented that reliably restricts access to the Science DMZ to cloud nodes that are running scientific applications.

The other security related aspect is implicit in the discussion above: will the “Science DMZ” model remain acceptable? Or will deep inspection of traffic flows be necessary? If so, with what implication for achievable bandwidths, site architectures and cost?

Treating the network as an infinite resource—the drawbacks

Video on demand streaming services do not treat network bandwidth as infinite. Instead, the application monitors network performance and adjusts video quality in an attempt to maintain an acceptable user experience. Of course, this is not always possible and on-demand videos do freeze. HEP applications, however, are nowhere near as sophisticated and the amount of effort invested to make applications network-aware has decreased the more the network is seen as being an infinite resource. Unfortunately, this means we are poorly prepared should the network ever become a bottleneck

Recommendations (Areas of Future Work)

WLCG should continue to play an active role in network provisioning discussions, and maintain strong engagement with NRENs, GEANT & ESNet on capacity provision and management, champion OXPs, and broker discussions with cloud service providers.

As HEP is one of the most demanding user of Research and Education networks, it will be of mutual benefit to keep, and increase the current level of engagement of the HEP community with (N)REN providers. Regular meetings among Experiments, Sites and Providers have proven to be very useful to better understand requirements and available features; such events should be organized in the future on a more regular basis.

Major HEP facilities should be encouraged to support WAN network development with the offer of colocation space in a neutral environment that can facilitate interconnections.

Networks and monitoring tools can provide real-time information about the healthiness and utilization of the network itself. This information can be used to improve the user experience and the utilization efficiency, if it can be retrofitted into the network by an intelligent system that can understand both network status and user expectations. The HEP community should investigate the possibility to develop such a system with the ability to work in a multi-domain network environment.

Overlay networks

Use of commercial cloud resources only accessible via the public Internet may require the use of overlay networks to securely attach such resources into the HEP computing system. The HEP community would greatly benefit if such connections could be done using standard protocols and mechanisms, independent from cloud and network providers. The HEP community should promote the development of a standard way to securely and efficiently connect remote cloud resources without compromise on performance and accessibility.

Other overlay networks such as LHCONE should be kept to increase security and fair use of resources.

End-to-end transfers

Both ATLAS and CMS expressed interest in end-to-end networking capabilities and both would like to understand if and how it might help with their ability to more effectively prioritize and utilize distributed resources. The term “end-to-end” transfers encompasses a range of capabilities involving data movement from source to destination and may involve one or more source or destination servers, special network configurations (virtual circuits, traffic shaping, traffic policing, etc.) and/or special engineering of the network path(s) used to optimize specific data transfers.

IPv6

Technical and operational limitations due to the scarcity of public IPv4 addresses would be suddenly lifted if IPv6 could be used consistently by all the Internet users. Thankfully, IPv6 adoption is progressively taking over IPv4 utilization and will free capabilities and resources currently wasted in dealing with IPv4 addresses shortage. The HEP community is urged to adopt IPv6 as soon as possible in transport networks and client-server applications: the exhaustion of public IPv4 addresses is going to impair the growth of the Internet and the development of new applications as time goes by; the HEP community must make sure to be ready for IPv6 well before the IPv4 ship is suddenly abandoned by network suppliers and operators in the coming years.

HEP has ongoing effort in this area in the form of the HEPiX IPv6 Working Group (http://hepix-ipv6.web.cern.ch/), formed in April 2011 to consider whether and how IPv6 should be deployed in HEP (especially for WLCG). The working group has continued for much longer than was originally foreseen and has made significant progress in identifying and addressing problems for IPv6 deployment for WLCG, yet challenges remain.

Data placement

The intelligent use of the available network capacity could assist in optimising data placement, and traffic routing. Current network usage patterns contain lots of bursts (short peaks over 90 percentile) which is a range that (N)REN use to plan their upgrades - increasing link utilization towards the peaks must take this into account. The intelligent use of the available capacity could increase throughput on existing links by factor 3-6.

In the long term, potential areas for intelligent use include network scheduling and end-to-end path selection with segment routing being one of the technologies that can create "programmable" networks over legacy MPLS-enabled links. Investigating technologies that could help bridge new and legacy networks will be crucial to enable adiabatic changes to the infrastructure.

Network-aware compute (workload management) will be needed to plan transfers in advance and could help synchronize different clients/experiments based on a backend SDN technology (this also brings an opportunity to create synergies with other R&D domains). Some early prototypes have started in both ATLAS (Panda) and CMS (HTCondor IO scheduling), but a lack of available SD-WAN infrastructure makes it difficult to progress quickly.

Technical Aspects of Future Networks

Network is a critical part of HEP infrastructure interconnecting site's resources and is a key-enabler for all the computing models described in [ref computing models]. Networks currently used by HEP are provisioned and operated in close collaboration with International and National Research & Education Network providers (REN/NREN) such as GEANT, ESNet, Internet2, etc. [ref to doc listing all of them]. NREN support for LHC programme has been key to its success and a crucial aspect of the network evolution for HL-LHC will be a forward-looking technical engagement with all the major (N)RENs as changes in the HEP network flows can have significant impact on all users of the network. At the same time, (N)RENs will face multiple challenges while trying to ensure the estimated growth rates are delivered including funding, complexity, vendor equipment timelines and deliveries, speed and evolution of the market as well as available space and power.

Some of the key technical aspects to be considered for possible computing models that could have significant impact on the networks supporting them would include apart from network capacity also structure of the network flows, i.e. large number of small streams vs small number of large streams, congestion protocols and their trade-offs, sensitivity to interruptions, packet loss, packet ordering, re-routing, changes in latencies and burstiness. Finally, location of the endpoints, the way they're interconnected as well as potential new network technologies will need to be taken into account to ensure optimal flows and available capacity.

Any distributed large capacity storage facility needs high bandwidth network connectivity to fulfill these main functions: *inbound connectivity*, for data recording from the data sources; *among storage facilities connectivity*, for internal data movement for optimizations and replications; *outbound connectivity*, for data distribution to remote computing facilities. While it may not be necessary to dedicate links to different functions, special attention must be given to the amount of bandwidth and the level of security applied to the different types of connections.

DCI - Data Centre Interconnect

Network technologies have been growing at a pace that should allow the implementation of a distributed storage facility of the capacity needed by WLCG. 400 Gbps connections are already in the pipeline of network manufactures and infrastructure providers, higher speeds are being explored while prices of older technologies decrease at usual pace. It can be envisaged that Terabps links will be available in the second half of the next decade at an affordable price. In term of technology, there is no major impediment foreseen.

However, cost and availability of long distance fibres vary considerably depending on world regions: west Europe and North America have a large footprint of high speed fibres that allows relatively cheap connectivity among countries in any of the two regions. While East Europe is catching up, a different situation is registered in Asia, where most of the countries have very good connectivity to North America through trans Pacific cables, while connectivity between Asian countries is hindered by complex political relationships.

In addition, an important factor for connecting Asia and Pacific (but also India, Pakistan, Russia and South America and Africa) is latency. Depending on the computing model, capability to execute end-to-end transfers over mid to high latency links is necessary to achieve effective use of resources in those regions and will require end-to-end tuning and optimal path selection

Today CERN has distributed its disk storage capacity between two data centres: ~150PB in Geneva (CH) and ~100 PB in Budapest (HU). The two data centres are interconnected with three 100Gbps links, which get saturated now and then. Assuming to have a dozen of large storage datacentre hosting hundreds of Petabytes each, it may be necessary to foresee WAN connectivity of at least 1 Tbps to each datacentre *(rule of thumb: 200 Gbps per 100PB of storage?)*

WAN - inbound and outbound connectivity for data computation

Being the data sources directly connected to the CERN data centres, the feeding of data to the storage centres of the Data Lake will be accomplished mostly via DCI links.

WAN connectivity will be used mainly for distribution and reception of data to/from the computing facilities. Each storage facility should provide enough WAN bandwidth to serve all the requests.

Today CERN has a potential of 600 Gbps WAN connectivity, counting LHCOPN, LHCONE and NRENs peerings, which utilization has peaked at ~270 Gbps during the summer 2016. It can be envisaged that 1 Tbps of WAN connectivity will be needed at each storage facility.

LAN (Data centre/Internal)

Responding to the challenges in the computing models will require a co-design of network, computing and storage infrastructures, which will have strong impact on the network architectures of data centres. The evolution will require moving from a static distributed infrastructures into systems that would depend on an efficient interplay of software with elastic and diverse set of resources in terms of compute, storage and network.

This will require embedded integration of novel methods of steering traffic to multiple locations to enable end-to-end transfers, novel modes of propagating information on data availability, network utilization and optimal path selection as well as greater interaction between end users and networks. To implement such systems it will be necessary to take advantage of the synergy between programmable networks and worldwide distributed systems interconnected by high capacity links developed for the data intensive research programs.

Network functions virtualization (NFV), which enables creation of virtual networks, routers and enables programmable network with dynamic routing decisions is a set of technologies - sometimes also referred to as network operating systems or software defined networks - that will be important to evolve towards multi-service multi-domain network systems as it will make it possible to create on demand network topologies (both within and between data centres), advanced bandwidth allocations, scheduling of high throughput transfers, establishing on demand routes for end-to-end data transfers as well as providing higher level of traceability and separation between different domains thus improving the overall security.

Software defined networks will also provide an important evolutionary step in the data centre design by providing open source technologies in areas that were until recently fully dominated by network vendors. This could potentially have a similar far reaching effect as the the recent migration from physical to virtual machines and container technologies that has radically changed the way we design, develop and deploy software. The impact for scientific workflows will require the management systems to be deeply network aware and be able to reactively and proactively respond to instant feedback on actual versus estimated task progress, state changed of the network and end systems. This will in turn enable better synergy between data intensive research domains to develop a new more efficient mode of operation based on software-driven bandwidth allocation, load balancing, flow moderation and on the fly topology reconfiguration where needed. The overall results should be an optimal use of the available network, computing and storage infrastructures while avoiding saturation, blocking and competition over the existing network capacity.

Security - Science DMZ

DCI links must be considered secure to avoid stateful/deep inspection of the packets they transport and thus allow the best exploitation of the available bandwidth. Access control can be applied only if it doesn’t introduce any degradation. To do this, DCI links must be used exclusively for the traffic exchanged among storage facilities. Data from a storage facility to a computing facility cannot exit via another storage facility.

WAN connectivity cannot be considered secure, so it is essential to provide the necessary control to avoid and detect intrusions in the storage facilities. This control should not penalize the legitimate users, though, thus some implementation of the Science DMZ concept must be foreseen.

Cost model

Unlike CPU and storage, network resources used by HEP are inherently shared with other R&D domains and directly benefit from the funding received by the national and international agencies supporting RENs and NRENs. The direct cost for the provisioning, operating and evolving network backbones used by HEP is thus borne by the RENs and NRENs with minimal or no direct cost borne by the experiments. It is therefore important to keep the existing synergies and ensure that R&E networks and technologies stay common across data intensive science domains.

There are also additional costs on the end-sites to provision, operate and evolve their network equipment and in certain cases, depending on the region, to pay for the uplink (“last mile”) connection to their closest NREN. Uplink costs have different ranges depending on the region, but in certain cases can be significant, as high as 1 EUR/1 TB transferred [ref PIC].

For commercial cloud resources, the cost for establishing and operating direct connections was shown to be significant in certain cases [ref Mike O’Connor BNL/ESNet direct Amazon experience], but in others, peering to RENs/NRENs was offered within the cloud provider’s premises thus greatly reducing the cost [ref. SoftLayer, T-Com]. This implies that the cost of connecting to the virtual private clouds could greatly vary depending on the provider, location of their data centres and availability zones. Significant challenges still remain in understanding how to interconnect multiple cloud providers within the REN/NREN framework that could impact the total cost of cloud networking [ref. NREN AUP].

Finally, some of the HPC centres are currently not integrated in the HEP and connecting them will require effort within (N)RENs, but it’s foreseen that the actual infrastructure investment in this case shouldn’t be significant as HPC centres are already connected with high capacity links.

Data Management and Data Access

Towards a common data management solution

The WLCG experiment data management systems are currently implemented with a multilayer architecture, as outlined in the following.

1) The fabric layer - consisting of the storage and network hardware at the distributed facilities

2) The service layer - implementing the storage systems and providing the network capabilities in the local area of the sites (LAN) and between different sites (WAN). The service layer implements also the security model with respect of authentication and authorization.
3) The protocol layer - defining storage, data access and data transfer protocols.
4) The data management middleware - consisting of clients and services for data access, data transfer and (meta-)data management.
5) The application layer - defining the high-level data management system, all experiment specific policies and practices, and the overall data organization

In defining the possible evolution of WLCG data management for HL-LHC, we might assume a future consolidation of storage facilities into a more limited, O(10) number of centres, interconnected with 10 Tb/s network links and serving data to O(100) processing centres with different connectivity (in the same LAN of the storage or remote, with connections ranging between 10 Tb/s and 10 Gb/s). We will not discuss the hardware layer, as the evolution of storage and network hardware 10 years from now is largely unknown.

The storage services and protocols in WLCG have so far developed and evolved in the context of high energy physics. Storage solutions such as Bestman, Castor, dCache, DPM, EOS, StoRM, xroot have all been instrumented with the Storage Resource Manager (SRM) interface. SRM is used for negotiating file access and transfers with the storage and implementing data and metadata management operations (such as removing files, staging files, looking up files and directories). All storage solutions also enable the gridFTP protocol, used for 3rd party transfers and file access. Each storage also offers a native protocol for file access, normally more performing than gridFTP for this purpose and also offering streaming capabilities. In recent years (LHC Run 2) the HTTP and xrootd protocols have been enabled at most WLCG storage systems and are being used by different experiments for different use cases.

In HL-LHC we might foresee a rationalization of storage technologies, where the variety of products implementing similar capabilities at similar scales might decrease in number. At the same time, storage evolution needs to consider technology evolution, such as object stores, as possible backends. Finally, storage services need to be easily accessible by non-HEP disciplines. Therefore, in preparation for HL-LHC, the community should engage in simplifying the protocol and storage ecosystem, leaning towards favoring non-HEP-specific solutions whenever possible, and seeking commonalities across experiments wherever feasible. In more detail:

a) The SRM interface, while offering an abstraction layer for storage management, comes with limitations, and HEP experiments have reduced their dependencies on it. Such dependencies should be eliminated completely, including its usage in tape management, for which a more lightweight solution should be considered. This step is important in allowing the experiments to benefit from new storage technologies without needing to implement the SRM interface to them.
b) The gridFTP protocol, while enabling reliable file transfer capability, is also relatively heavyweight in terms of authentication and authorization and relies on the WLCG security model (X509) which is per-se rather HEP-specific. Possible alternatives to gridFTP should be investigated, such as adopting a protocol allowing 3rd party

replication as well as file upload and download as well as streaming. Such a protocol should enable also file deletion and some basic metadata operations (e.g. stats). It is important that check-summing is also properly supported by the protocol, reducing the load on the storage itself. Multi-streaming has been a key component in WLCG so far, given the long RTTs between several sites; the functionality should be preserved.

c) In HL-LHC we foresee a reduction of the storage endpoints to large and reliable sites. Sites with CPUs only will play a major role in event processing and therefore, we will need to provide data to the CPU cores minimizing the lack of data locality. We may need therefore to develop a caching system that can use the WLCG persistent storage as a backend and provide Content Delivery Network capabilities for remote computing resources. Several technologies and implementations could be envisaged and some research and prototyping in this respect is highly desirable.

d) The storage technologies should evolve in a direction favoring consolidation and clustering. Examples today are the distributed dCache installations in NDGF and in the US and the EOS deployment between CERN and Wigner and in the Russian federation. Demonstrators in this area should focus in demonstrating the deployability of such solutions and addressing how best to organize the data in the clustered storage to serve CPU cores at the participating sites and at external sites.

The data management middleware layer on top of the facilities and protocols, could play different roles depending on where we decide to put the boundary between experiment-specific and experiment-agnostic functionalities. The FTS2 service did provide real minimal functionality on top of the storage services, while FTS3 brought in more intelligence and functionalities; still, a lot of the intelligence in scheduling transfers belongs to the experiments' data management frameworks. In this context, solutions that increase the level of commonality across experiments here should be investigated and prototyped. For example, any file transfer service solution could serve as the baseline of all asynchronous data management actions, including deletions, by incorporating more network awareness. Enabling features such as Software Defined Networks could be part of such high-level services rather than building such functionality inside the experiments' systems.

The data management systems in LHC experiments are today rather diverse, sometimes for historical reasons and the original lack of maturity of the Grid middleware; sometimes for the conscious decision to organize data and information in a way consistent with the operational model of the experiments and the culture of the community. Attempts to rationalize such models across experiments in view of HL-LHC should be seriously considered, and a conceptual design for an evolved, common high-level data management system should be evaluated. At a minimum, the experiment-specific layers should evolve to a more modular architecture, allowing incorporation of external components, developed in concertation with other experiments and communities. Key functionalities such as a flexible metadata system to store user-defined information are lacking in many experiments and there is an opportunity now to study and implement coherently new common services to be adopted by more than one experiment.

Finally, the new modular monitoring systems such as the WLCG agile monitoring offer a platform for the evolution of data management monitoring. A key point is that any monitoring should be developed coherently as part of the process rather than a-posteriori on top of some already developed systems in response to a need for such tools. The data management services would then be able to establish a solid and well-designed feedback loop with the monitoring infrastructure and leverage quasi-real-time information about the state of the underlying facilities, which may be crucial e.g. in fully integrated network-aware solutions.

Data management for fine-grained processing

Enabling fine-grained processing through serving data in smaller chunks, or by streaming data, will be crucial in enabling the effective use of new types of resource now being offered, and likely to become more important in future. Such resources include those that have no (WLCG-)managed storage system, opportunistic resources that may be available only for a short time, backfill resources where the payload may be pre-empted at short notice, and so -on. Being able to fill those resources efficiently means being able to process small pieces of data. It is also likely that being able to stream data efficiently to such resources will also force a deeper understanding of end-to-end data flows that will be of benefit in all use cases, even for larger data sets.

Fine grained processing introduces new requirements and challenges for data management. The standard data management (DM) paradigm is based on managing files, typically and preferably large for the sake of storage efficiency, aggregated into datasets, collections and so on, with metadata management at the same levels. Fine grained processing works with small chunks of data representing events or event collections, each chunk with its associated metadata. Even if the chunks are files this breaks the large file paradigm and demands scaling up the number of artifacts managed, and efficiently managing small files. Ultimately the storage efficiencies of large files do not go away (with possible exceptions such as object stores), but small files can be compressed or packed as necessary, which should be supported by data management and workflow management (and for ROOT formats is already supported for application data access).

Event streaming, quasi-continuous data flows supporting event service type workflows, also introduces requirements for supporting dynamic data flows in the DM system. An event streaming service (ESS) will include an intelligent CDN-like layer presenting a simple interface to the client for servicing fine grained data requests while behind that interface applying DM and workflow system knowledge to fulfill the request, together with any data handling, replication, caching etc. that the fulfilment might entail (see the data access section).

The ESS will employ cache hierarchies ranging from buffers at large storage repositories to cache-only stores at small sites to local cluster caches, and will rely on the DM system to keep track of what caches contain and to resolve the closest replica for a given client data access request. The ESS will further rely on an event or event range level catalog to manage metadata associated with fine grained data artifacts. Efficiency would argue for the fine grained aspects of the DM system being supported through this catalog, which could be part of the DM system itself if it was scoped to flexibly include metadata beyond that needed for DM.

Need for R&D

Such a service will require significant R&D and prototyping. This should result in a common service for all experiments to support the management of fine-grained data products, and the associated metadata, etc. There is some existing experience of some aspects of this in ATLAS.

Data Access

Data access is where the processing workflow and application payload meets the data management infrastructure. If data access is to be efficient across the wide variety of use cases it must be a strong driver for the architecture of both the processing and data management. The data access problem is largely one of minimizing latency, or the impact of access latency on processing workflows. Data access latencies come in several forms. Storage retrieval latencies can range from an SSD read time up to hours for a tape stage-in. Network latencies range from manageable if not transparent LAN latencies within a facility up to potentially debilitating WAN latencies of hundreds of milliseconds. Decompression latencies are a fact of life when storage costs make compression essential, and can be the dominant walltime component in CPU-light workflows. We describe here approaches and mechanisms to ameliorate latencies and achieve efficient data access amid the requirements and constraints of the complex distributed computing and concurrent software environment of the next decade.

With storage the largest facility cost component today and probably also in the HL-LHC era, minimizing the storage footprint of data on (particularly) disk is at a premium, driving data management towards minimizing file replica counts (to as low as zero in cases where generation-on-demand aka virtual data is cost effective) in favor of using disk as a transient cache for the data currently in play. Cost pressures and the efficiencies of consolidation will drive down the number of storage sites, a process that has already begun. Furthermore the nature of some facilities build in an inherent separation between processing and storage, such as cloud based CPU and HPCs utilizing HEP site based storage. A consequence of all these factors is that the average latency of data access will rise. This will put a premium on carefully measuring, optimizing and mitigating these latencies. At present, some latencies in the system are effectively human ones: the difficulty of fully automating data provisioning and access into HPCs in particular can result in manual steps appearing in the workflow to pre-place data. For the future, latency mitigation will be essential, by fully automating data access workflows on all resources, and within those workflows, by decoupling access latencies from processing workflows through mechanisms such as fine grained processing, described below.

The demands on data access differ depending on the nature of the workflow. For interactive processing, minimizing the time to deliver results to the user is paramount. For offline/batch production, maximizing the aggregate event throughput across all the tasks and resources currently in play is most important. Can these differing goals be achieved with the same optimization strategy?

One strategy that can benefit both is fine grained workflows. For the interactive case, partitioning the work performed by a single processing thread tunable to a few (N) minutes both allows results to be incrementally delivered back to the user in near real time, and offers a trivially parallel mode of operation to distribute the work over available cores. Maximizing the cores employed, if it can be done efficiently, is a means of minimizing the completion time. Elastically expanding the active core pool beyond the local machine and into a local cluster or remote cloud, given a workflow system able to support it, becomes a practical possibility if the data required is dynamically streamable and this promptly available thanks to fine grained delivery.

For the production case, keeping processing resources fully and efficiently busy requires keeping them continuously provisioned with input data. The dilemma of matching tasks, processing resources and the task input data given the heterogeneity among all three -- task resource usage signature, diverse processing resource capabilities, and data dispersed across storage resources -- is one of the great challenges of automated distributed workflow management. The more constraints there are on the system, the more difficult and costly (in terms of inefficiencies) the matchmaking is. In the coarse grained workflows of today based on input files that are typically large to optimize storage performance, data locality is a constraining requirement on the system because of the long latency in replicating a large file to a processing site, or in the direct access case, the processing inefficiencies incurred by WAN access latencies. A fine grained workflow on the other hand can make it practical to employ both direct WAN access and dynamic real-time remote transfers, and the two in combination.

A fine grained ‘event service’ workflow partitions the processing, and the attendant input data accesses, into short (e.g. 10 min) chunks, and decouples the payload processing from the provisioning of the input data chunks by performing data access asynchronously from payload processing, with input data chunks staged to and cached on the WN, ready for local worker consumption. In a scenario where the processing resource matching the task’s requirements is remote from the input data, the processing can commence immediately, with the payload initialization running concurrently with the retrieval and caching of the first input chunks needed. By the time the payload is initialized the first inputs are likely to be locally available, and event loop processing commences with little or no latency incurred from the remoteness of the data. Within the event loop, a data access agent in the worker operates asynchronously from the processing to progressively retrieve input chunks, caching them locally for worker processes to consume with no WAN latency. With this workflow, a task may be assigned to a processing resource far from its input data, with processing commencing immediately without a pre-staging step, and without inefficiencies due to WAN latencies.

A further optimization within this fine grained workflow may be to perform ‘lazy caching’. If it were beneficial for resource and network usage to do so, a decision to assign a task to a processing resource remote from the input data could be accompanied by a directive to replicate (cache) the inputs close to the processing, e.g. if it were known that the probability of reuse were high and it was beneficial to offload the remote traffic from the network and the data source. The worker would initially be obtaining input chunks from the remote resource, but once the local replica was available, it would transition to using the replica. Other consumers of the data would directly utilize the replica.

In coarse grained processing, file replications are performed with no consideration of what data within the file is actually going to be used by its consumer(s). Given that sampling fractions of 10% or less are not uncommon, this can result in much more data being moved across the network than is really required. Fine grained processing, in which data accesses are associated with a particular task and thereby with particular and known data requirements, offers an opportunity to tailor the data moved to the data actually required, by employing a ‘marshalling agent’ close to the data. The marshalling agent could access the data from storage (including staging from disk if required), perform any pre-determined event filtering that is defined by the task (e.g. applying a trigger mask), select only the event components (branches) needed, and cache the data server-side ready to be delivered as chunks to workers (and, also available for lazy caching). This data preparation and marshalling can be performed in advance, before the associated task is ever dispatched for processing, since the workflow manager knows it will be needed as soon as the task is defined. It can also be performed concurrently with the initial processing, ‘lazy marshalling’ akin to lazy caching, except when staging from tape is part of the marshalling.

All of these fine grained processing scenarios presume the existence of an intelligent agent with which a job’s input data access process is communicating. This agent serves data to the job accordingly for what is available: direct read to a remote file, read from a locally cached file, retrieval previously marshalled chunks, and so on. This agent is the ‘event streaming service’ (ESS) and can be progressively elaborated with more intelligence and capability as it evolves. It’s key attributes are that to the client it presents a simple and transparent means of accessing the data requested, while on the server side it is able to communicate with and apply the intelligence of the full workflow and data management systems to optimize how the requested data access is fulfilled.

The ESS will have multiple components, in its more developed forms. In its simplest form, it would provide simply a data access URI for the requested data, obtained through interaction with the data management system, to direct the client to the ‘closest’ replica of the desired data file. More sophisticated versions of the ESS would collocate ESS agents at data repositories to perform data preparation/marshalling under the direction of the workflow management system, and the ESS service in communication with the client would then direct the client to the marshalled data, again via the data management system so long as the marshalled data is cataloged by the DDM, which has advantages in uniform data artifact management, ability to distribute marshalled data to caches, manage its deletion and so on.

What measures can mitigate compression latencies in data access, given that there are workflows for which this is a significant proportion of the processing time? One measure is to take advantage of workflows that isolate data access from payload processing, like the event service, by performing decompression within the asynchronous data access process. At the same time the input data chunk is retrieved, it is decompressed, so the chunk read by the payload worker is already decompressed. Also for centralized caches, when/whether to decompress as part of the data caching/preparation process can be considered. For long-lived data the (CPU) cost to (storage) benefit relation will always favor compression, but depending on the data lifetime and how it is used, a particular cache replica may or may not be best kept compressed. A short-lived intermediate data product may best be uncompressed, whereas a data chunk that is to be moved over a WAN may be best compressed even if its lifetime is short. We should ensure that our data handling workflows include the ability to make informed decisions on when to (de)compress, and provide for the metrics gathering and cost/benefit calculations to drive those informed decisions.

For mitigating tape staging latencies, and generally optimizing the processing, storage and latency cost of bulk processing across a defined sample of data, train-style processing is a proven technique that we expect will persist and become more prevalent. Train processing aggregates the processing steps to be applied to a defined data sample, provisions the required input data, runs the aggregated processing against the input data and performs the whole in the context of managed production workflows. The input data provisioning step can include staging from tape, and/or optimizing the brokering of the processing across distributed resources to maximize data locality. Train production makes the most of the data read, serving many clients the same data stream and reducing the inefficiencies of individual clients consuming a stream for which the sampling fraction may be small and the cost of provisioning the data substantial.

Non-volatile memory may in the future allow to exchange data between processes in internal (RAM) shape rather than externalised (disk) shape. This could potentially reduce the CPU path required to re-access the same data, such as large and often reused field maps or other conditions/calibration data, but could also apply to data access down the processing chain of train production. This would likely need support in the data management and data access layer which do not yet exist.

Commercial cloud computing is of growing relevance for HEP and LHC computing, a trend which not only will continue but could become a major part of our computing model. Over the past several years the cost differential between adding compute cycles via Amazon AWS vs. procuring owned equipment has shrunk from ~7-10 to ~1.5 by some estimates. Even conservative extrapolations of the trend would ensure a prominent place for commercial clouds in our computing on an HL-LHC timescale. Usage modes and workflows must be carefully designed to minimize costs, and this applies particularly to data access and management. Data products stored for appreciable times translate directly to costs and weaken the cost/benefit for a given processing throughput. There will be a premium on maintaining a light data footprint, e.g. minimizing intermediate data products. While data ingress is free (at least today), egress carries costs and so will drive exploiting cloud-generated data within the cloud as much as possible before data egress to owned storage. This argues for managed train production within the cloud, minimizing the lifetime of intermediate data products if they exist and making maximal use of the data to produce compact analysis formats that are relatively cheap to export.

Need for R&D

Again there is an urgent need to start prototyping such solutions:

- Some workflows access only small fraction of their input data files: both event service and train approaches promise to reduce the resulting inefficiencies
    - estimate overall distribution from wlcg production
    - compare both improvement approaches taking into account the dependency / fan-out factor if IO is amortised across more than one job

- cloud processing + private storage approaches will put a price tag on externally stored intermediate data products
    - can the job-group deployment granule be optimised to reduce transfer costs (existing experience from commercial deployments?)
    - same saving might also reduce medium term storage volume

Resource Provisioning

Resource provisioning in the context of this paper deals with the problem of getting workloads (jobs) running on compute resources provided by WLCG partner sites, and other opportunistic or commercially provided resources. It also addresses the area of local site management of resources (e.g. batch systems), but of course for many opportunistic, commercial, or other offered resources, the batch system or interface to submit work is not under our control. Thus we must consider a resource provisioning service that is adequately able to adapt and interface in all such circumstances.

On the timescale of HL-LHC, the scale of the infrastructure is expected to significantly increase; some 20 million cores will be required to fulfill the CPU needs of the LHC program, which represents a >20-fold increase in scale of job slots, etc; and if we are dealing with smaller processing chunks (see previous discussions on event streaming and data management) we could be looking at >>100-fold increase in processing objects. It is clear that this increase will not primarily come from an increase in the number of WLCG sites, but rather via increasing scale at existing sites, and significant contributions from other sources where there is no control over how those are presented to us. In addition, it is highly likely that other large scale science experiments will be sharing many of the LHC computing centres and have equal influence over how they are configured.

Thus, our community needs to deal with a larger-scale infrastructure composed of many sites over which we (at the infrastructure level) have less influence compared to today. Today, dealing with the complexity of the infrastructure due to nonuniformity in configuration is a significant cause of operational effort. Until this point, we have tried to mitigate this by aiming for uniformity as far as possible; the major components are CEs and batch systems, but there are numerous other sources of effort in the details of the OS configuration. So far, our community has been successful in at least mandating a specific version of the operating system.

This situation will change for the reasons noted earlier. Isolation of the “WLCG world” and “site world” is necessary to avoid further complexity. The use of containers or VMs are obvious candidates for achieving such an isolation; furthermore they provide the potential for worldwide uniformity of an experiment’s computing environment, immensely simplifying deployment and debugging for the experiments.

Such a strategy could simplify the computing environment: for example having experiment-specific standard containers, which are versioned. This would solve a concern with data provenance; one would version the container, not every single library on every WN system including compilation and other options. The software environment would be the same across everywhere (via the container) - same OS, same installed software, same paths, same configuration, regardless of how a site is configured. Such a standard container could be used on any type of resource (Laptop / desktop, Tier-3 type local analysis cluster, Cloud cycles, WLCG infrastructure, HPC resources).

Provisioning

There are several options for starting the containers and getting experiment jobs (at whichever granularity) running.

1. Continue essentially as today, except the jobs submitted by an experiment to a site are basically “run this container”.
2. Give experiments a secure “push button” interface, “create N instances of containers with a certain set of parameters (memory, network, lifetime) and version V.
3. A site schedules, according to their own policy, with information from the experiments, “jobs” that start experiment containers on behalf of the experiment.

The idea in all three cases is that the container, when started, should know what to do from that point on, and the site has no knowledge of what is going on inside. Starting a container is taking the place of starting a pilot job, the pilot job needs no further instruction from the site, it knows where to get its work.

The first option 1 is the least work but has almost zero benefit; the experiment still needs to deal with both pilot factories and with multiple CE flavors and possibly still with different batch system flavors. Option 2 is a bit more work (but a prototype already exists) but reduces the CE component to only what is needed. However, this does not allow the sites control over submission rates.

Option 3 is perhaps optimal for both experiments and sites, as the sites are responsible for getting the containers running and balancing their own workload between the various groups (different HEP and non-HEP user groups), and the experiment is responsible for everything that happens once the container starts (eg pulling work from a remote task queue). A site’s container scheduling factory could include a rate limiter / start rate backoff that is tuned to whatever is appropriate, and allow a site to manage potential load-related problems.

A further advantage of such an option, is that it can reduce the experiment-specific support needed at a site. In order to start a container, the site needs a relatively limited set of information from the experiment, details the location of the container, how many to run, etc. If all the experiment-specific work is happening inside the container, these people are not needed, a container in Lyon is identical to a container in Amsterdam.

One obvious area for common effort across sites would be to look at having a common CE and potentially batch system layer; this would reduce significantly the global debugging and development effort needed to run these services at sites and would allow WLCG to share operational experience more readily. It would additionally provide the experiment provisioning frameworks a single API to use which would substantially reduce debugging and ongoing development effort. Assuming a common product was chosen, it would need to

- be simple to configure;
- be open source and free to use;
- scale for both large sites and small sites;
- be able to integrate with a number of different batch systems easily

The HTCondor CE and the HTCondor batch service would seem like an obvious choice for this common layer. Both the HTCondor CE and batch services are comparatively easy to configure for small to medium sites and the source is freely available with the software free to use. The current CMS global dynamic pool is currently known to scale up to 200k running jobs and CERN is currently operating a local cluster at 160k cores.

HTCondor is normally configured to manage a pool of HTCondor nodes locally, but in the case of a site having resources that are provided by a different batch system (for example by the local computer services department who mandate the batch system choice), HTCondor can be configured, via a library of GAHP providers, to submit work into that batch service. GAHP providers exist for most, if not all batch systems in use in WLCG today. Providers also exist to provision dynamic capacity for a site on Cloud Resources, notably Amazon Web Services and Google Compute Engine, etc. HPC resources can be backfilled the same way, typically using a suitable GAHP (e.g. for SLURM in the case of many HPC sites).

Such a service would also thus address the need to be able to schedule work a wide variety of resources with very different local services, configurations, and management policies.

Common Fabric Services

While the discussion above tends in the direction of the experiments being able to make use of any resource, no matter how it is locally managed or configured, there is still the need for supporting WLCG sites that need to follow a general recommendation for how to setup and configure.

With the diversity of workloads already increasing, it is expected that the relatively homogeneous configurations of today will become diverse. The use of combined compute/data platforms such as Hadoop blur the simple distinctions of disk or compute servers. Ongoing work with Intel's Rack Scale Servers is expected to allow rack level resource allocation which will reduce the 'Tetris' scheduling problems to fit workloads to the appropriate configurations. It will however complicate the resource provisioning systems as hardware configuration becomes software managed in cloud orchestrators or batch schedulers: having a minimum number of batch schedulers within WLCG will permit us to explore common solutions to this problem.

Common configuration management, ideally with only one or two of the main standard tools (e.g. Puppet, Ansible, Chef, Salt) is a goal that WLCG should aim for. At the very least, pools of experts to extract and maintain genuinely common configuration templates should be encouraged within WLCG, and we should have a "standard recommendation" for new sites that just want a reasonable default.

Minimising the "amount of things" a site needs to configure can substantially reduce the risk in divergence of configuration management systems, so a more distributed model where the "tricky parts" are configured regionally is favoured, leaving the sites the configure just the basic hardware and just enough to join the regional instance. Which brings us to...

Regional Consolidation

Consolidation of resources within a country or a region is something that is already happening, in order to reduce operational costs and support staff, and to benefit from opportunities related to larger scale centres. In addition, as noted above, it is expected that in some cases, regional academic computing resources might be supplemented by commercially procured resources (clouds) or with contributions of a National or regional HPC centre. In these cases it is far more effective for the global collaboration to consider a regional infrastructure as the contribution to WLCG, via straightforward and adaptable interfaces, rather than having to deal with each regional idiosyncrasy. Such a model would still apply to individual sites, but would also allow a region to manage internally how it provides resources to the experiments.

The benefit to the experiments and to WLCG would be fewer independent entities to deal with, thus simplifying some of the management tasks. Clearly the interfaces to such entities need to be defined clearly. The regions would be responsible for delivering to the pledges as now, but also for correctly reporting accounting and usage statistics.

In such a regional model, a single centralised (or several disjoint but centralised) matchmaking instance(s) and CE entry point(s) could be operated by the regional team - and the sites provide pools of resources which can then join the regional instances to accept work.

This model potentially saves significantly on staffing, since the local site then only needs to provide the hardware management and a system to spin up a standardised “box” (whether container or VM) that is able to call back and join one of the centralized instances. The model also centralises the responsibility for the region’s accounting.

A number of potential technologies exist for sites to spin up these “boxes”, for example:

- VM technologies, whether OpenStack (if the site already uses it) or light-weight VM provisioning tools such as Vcycle.
- Container provisioning technologies, of which there are many - Kubernetes, Mesos, Fleet, Docker swarm, ...

The choice should be made to reduce the operational effort of the site, since many of the potential solutions do far more than is needed for the simple case of spinning up “another box like the last one”. Combinations of CVMFS and container technologies can reduce the need for the site to maintain full grid stacks and allow the experiment workloads to define their environment independent of the site as long as the base operating system and provisioning tools are at an appropriate level.

In terms of technology for a regional batch service, HTCondor would be an excellent choice for this as it has been shown (with CMS) to work extremely well with large, dynamic pools of resources distributed across the WAN. Via its GAHP providers it would also support sites that cannot directly or do not want to join the regional instance. To address any concerns of scaling, the regional team could equally operate several service instances located at major regional hubs.

We note that this model retains the current “site scheduler” flavour, albeit in a regional form, in the sense that the boxes being created would dynamically join a regional batch system, and subsequently accept experiment payload via that - so it retains the orthogonality of the “site scheduling” vs. “the internal experiment scheduling” - but at the expense of running a regional batch service. The alternative model, for example, from Vac, is that the sites just agree to create experiment pilot VMs directly on their local resources in the agreed ratio. WLCG should consider which model is the best for the longer term.

Finally, we note that, for both models, ceding control to a remote team may be a potential concern for many sites, though if the model can be shown to work by some regions prototyping it, the funding realities might encourage sites to be more flexible with its adoption. Common development of tools across WLCG to serve such regional federations would be a useful step in this direction.

Cloud Capacity

As noted above, for a site that wants to provision some of its resources dynamically on public clouds, HTCondor provides the facility to directly instantiate and manage worker nodes that will join the cluster and accept work.

An alternative approach, pursued by CERN, and more suited to provisioning flat capacity in the cloud, is to use normal cloud provisioning tools (many exist, CERN is currently making use of Terraform) and normal configuration management tools to create and manage worker nodes in the cloud. Any cloud that supports the standard provisioning libraries can be made use of in this way. At CERN, the new HTCondor machines then call back and join the cluster as normal worker nodes. This model would be suited for larger sites wanting to provision some of its capacity on clouds or for regional teams wanting to provision some of the region’s capacity on external clouds. The latter probably fits better with regional funding models (i.e. it is the regional funding agency which buys the cloud capacity, not the local sites).

In both cases, it’s normal to provide a specific route (or an extra CE) via which cloud-suited work can be submitted by the experiments.

Opportunistic Capacity

Opportunistic resources are increasingly being made use of by WLCG experiments. They fall into two rough categories:

- **“No SLA”** where there is no expectation of any level of service or level of performance. In this category would be systems like BOINC, where jobs run purely on volunteer resources with typically no SLA and no significant networking resources.
- **“Mid-SLA”** where the resources are typically more uniform and reliable, the networking is often better. A larger range of jobs should be able to run on mid-SLA resources. Examples are:
    - borrowed (“backfilled”) HPC resources;
    - the batch-on-disk-server prototype work going on at CERN;
    - filling “service headroom” on an OpenStack cloud where unused capacity for services is used for batch workloads until requested.

- Related, though not strictly opportunistic, since it is paid for, is cloud spot-markets. Here, a standard machine ("mid-SLA") is available, but liable to be taken back at a moment's notice.

The distinction is somewhat blurred, for example, by running natively on large HPC clusters without any outbound connectivity (or for example, access to CVMFS): the performance is typically very good, but special measures need to be taken for the job stage-in and out. WLCG should study and publish common strategies or templated solutions for these type of resources.

In both cases of SLA ("None" or "Mid"), the workload is liable to be pre-empted and the workload management system of the experiment needs to be able to handle that.

In terms of technology, HTCondor is once again a good bet: it is able to manage and integrate heterogeneous resources very easily. The LHC@Home project resources are made available via an HTCondor instance using the BOINC GAHP. Having a single submission point with HTCondor and multiple different pools, also allows flexibility in the way in which jobs are served: e.g. jobs for the LHC Sixtrack application are submitted to the central batch service, served by standard batch resources from the share of the relevant CERN department and also served by BOINC resources.

Pre-emption and no-notice termination

Critical to the maximisation of opportunistic potential is the ability of experiment frameworks to handle robustly cases where the resource owner would like to get the resource back quickly. There are two potential cases:

- A job is "pre-empted" with a signal and has ~30 seconds notice to terminate, enough perhaps to notify a remote bookkeeping catalogue of the kill. This could occur in the case where the job is back-filling some resource; it is typically a function provided by most batch schedulers. In this case, WLCG would need to agree what that signal is, such that experiment job frameworks can make use of it. This typically can be provided on HPC or other batch-system backfill.
- A job is killed with no notice or signal. This is the likely scenario where the resource being used is not a batch system, e.g., a cloud service, cloud spot, or where the capacity needs to be returned quickly in case of problems, e.g. the disk-server case.

Experiment frameworks that can robustly support the second case (kill with no notice) will be in a position to maximise their use of opportunistic resources, so experiments should be encouraged to move their frameworks in this direction. The earlier discussion on event services and event streaming will help in such a situation.

Common Pilot Factory Framework

The arguments made for having site or regional batch services using common technology can equally be applied to the experiment workload systems. Since the scheduling problem is rather similar (and orthogonal) to the site scheduling problem, it makes sense to use an existing batch-like solution for this.

If HTCondor could be agreed as the common experiment scheduling layer, sites, regions and experts whose staff support both the site and the experiment would gain, having only to learn and debug one scheduling system rather than two, albeit in two different contexts (as CMS have found in OSG). HTCondor also integrates rather well with itself, so cross-system debugging is made easier (e.g. information from the experiment HTCondor system is automatically propagated to the site's HTCondor system).

Even in absence of a common site/regional scheduling layer, a common pilot (or container) factory framework would bring significant manpower savings across the experiments (and the sites that support them) once the initial investment to adopt it had been made.

Prototyping

The most interesting demonstrator would be to prototype the distributed operations of a regional HTCondor instance with several sites, accepting jobs from multiple experiments.

The demonstrator should examine:

- The lightest possible way(s) of instantiating "boxes" on a site to provide resources to the regional HTCondor instance.
- How the regional team can monitor and operate the resources across the region and the required operations model between sites.
- How to connect the regional instance to an existing site (i.e. via that site's CE) to allow the integration of sites that provide traditional Grid connected batch systems.
- Understand how the regional accounting can be handled, notably ensuring that appropriate credit is given to the sites that have actually provided the resources.

Workload and Workflow Management

Workload management (WM) systems orchestrate the execution of processing workloads across the computing resources available. Over the last 10-15 years WM systems in HEP have become very sophisticated as they have grown to serve a broad range of production and analysis workflow types on worldwide resources; as they use diverse resource types from the grid to HPCs and clouds to volunteer computing; as they provide data provisioning and access strategies; and as they enable the managed resource sharing needs of large experimental collaborations.

Workflow management is a closely related, higher level function that encompasses workload management and extends it to cover the entire processing workflow from the definition or injection of a processing task by a client (such as a physicist or physics working group), to its scheduling for processing amid constraints such as priority and availability of suitable resources, its execution by the WM system in concert with the requisite DM for data provisioning; delivery of results and/or notifications to the user; and the monitoring and bookkeeping of the whole.

One driver for the growth in sophistication of WM systems has been the importance of leveraging opportunistic resources. In order to keep operational needs manageable, integrating such resources should be transparent to the experimental and user communities. This puts the onus on the WM system to make all resource flavors look and function as uniformly as possible, as seen by system operators and by the work allocation and brokerage components of the WM system itself.

WM systems have most often been developed as fairly deep systems at the experiment level, above a foundation of middleware tools and services. While developed initially for a single experiment, systems such as DIRAC and PanDA were later generalized for use by other experiments. Today there are few if any highly developed WM systems dedicated to a single experiment. In the underlying middleware there is even greater commonality: most WM systems use common middleware components such as HTCondor. This is a trend we can expect to see continue as high quality solutions evolve and mature - a trend we should foster, while recognizing that this cannot be achieved by imposing external solutions on experiments irrespective of quality and suitability to requirements.

An architectural common denominator that has emerged among WM systems is that of 'pilot jobs' in a resource provisioning layer. In supporting a broad assortment of heterogeneous resources on a distributed fabric, bringing an appearance of uniformity to the resources as seen by the WM system has great value in controlling complexity and promoting robustness. Pilot jobs provide such a layer of uniformity between the physical resource and the WM system, serving as an insulating layer that encapsulates the complexity and specificity of a processing resource and presents it via a uniform interface to the WM system. Pilot jobs can also validate an acquired resource such as a worker node (WN) before presenting it to the WM system as an assignable worker. Because pilots need not be (although they may be) assigned a specific workload until they are launched, validated and communicate to the WM system, they can support 'late binding' of workloads, with the WM system retaining control over workload assignment until the 'last moment' such that dynamic decisions in response to changing circumstances like prioritization and resource availability are possible for as long as possible, avoiding high priority work being 'trapped' by long waits in busy or broken queues.

There are a number of additional challenges in the horizon which will make the management process more complex:

- Integration of diverse CPU architectures, like GPU's, ARM, powerPC, in an effective way in order to make best use of them and to optimise throughput;
- Integration with evolving data and storage models;
- Active control of networking, or adapting workflows to changing network environments;
- The ability to incorporate new workflows like machine learning.
- etc.

Fine grained workload management

Historically and currently, processing typically reflects the granularity of input files, which tend to be large to optimize storage performance. This coarse grained processing is reflected in workflows that first provision or matchmake the data needed for processing, then run jobs with running time counted in hours, then move the output files as necessary to a secure accessible location. An alternative model in which the processing is fine grained -- with parcels of work a few minutes in length -- presents many opportunities for making workflows more flexible and efficient, particularly in opportunistic processing environments, and where data and processing may be far from collocated.

Benefits for opportunistic processing: Opportunistic processing is typically characterized by worker lifetimes that may be short and/or unpredictable. Examples are commercial cloud spot markets, HPC backfill, grid sites and clusters subject to preemption, and volunteer computing. In these cases, fine grained workflows that can quickly begin to exploit a worker and quasi-continuously extract completed work during execution, such that a terminated worker engenders only small losses (the few minutes of processing time of the current uncompleted chunk), can bring substantial gains in utilization efficiency. Fine grained input data fetching delivers enough data to a new worker to allow it to quickly begin processing. Quasi-continuous input fetching asynchronous from the processing keeps the job provisioned with data, without pre-determining the total job length, so the job can elastically consume work for the lifetime of the resource. Quasi-continuous uploading of fine grained outputs from the worker to a storage endpoint gives real-time information on the health and productivity of the worker, and should the worker vanish, ensures that only the last few-minute parcel of work is lost and must be reassigned to another worker. Should an event fail, that parcel is reassigned to another worker and the job continues; a bad event doesn't bring down the whole job and waste the processing.

Benefits for distributed data: Today and in the future experiment data is spread across many sites, as is the processing. Storage costs make it advantageous to minimize replicas. Consequently it is often the case that the processing resource available for a task is far from its input data, which degrades efficiency due to WAN latency and the reduced robustness of a persistent WAN connection. Many strategies exist for dealing with this, such as constraining the processing to happen close (in network terms) to the data location, pre-placing data at the anticipated processing site, selectively increasing replica counts for popular data, and so on. Fine grained workflows present the opportunity to serve workers with the data they need without suffering the inefficiencies of non-local data. As the worker receives its quasi-continuous stream of fine grained work assignments, it can retrieve the associated data from source asynchronously from the processing. The data parcels are small and can be retrieved quickly even if far away, and the probability of successfully completing the retrieval is high. The parcels land as small files in worker cache (and can also be uncompressed asynchronously), ready for processing.

Workload management and analysis

Currently analysis is supported in workload and workflow management systems in a batch-like mode of submitting work to 'the system' and waiting for results to come back. Development and interest is increasing in 'Analysis as a service' (AaaS) type approaches, implemented for example through interfaces that leverage highly interactive components like notebooks. An interesting R&D area to explore, identified at the May 2017 HEP analysis ecosystem workshop, is the convergence of these approaches. Automation of workflows through the leveraging of workflow management systems as general analysis services can be an approach to robust analysis workflow engines that automatically/transparently recover the problematic last 1%. AaaS style interfaces backed by a workflow engine could be a means of elastically scaling interactive analysis at the back end from a local cluster to a large grid.

Analysis Facilities

Analysis needs to be fast and interactive, facilities for analysis need to minimize the “time to insight” and provide the analyzer with sufficient interactivity to enable a deep and faceted dialog with the data. Many analyses have common deadlines defined by conference schedules. The increased analysis activity before these deadlines require the analysis system to be sufficiently elastic to guarantee a rich physics harvest. It is clear that heterogeneous computing hardware like GPUs and new memory architectures will emerge and can be exploited to reduce the “time to insight” further.

The baseline analysis model utilizes successive stages of data reduction, finally analyzing a compact dataset with quick real-time iteration. Experiments use a series of processing steps to reduce the large input datasets down to the size suitable for laptop scale analysis. There is not a consensus on where the line sits between managed production-like analysis processing and individual analysis.

An evolution of the baseline approach is the ability to produce physics-ready data from the output of the (second stage) high-level trigger of the experiment. Once the data is prepared for analysis by completing most of the CPU-intensive processing steps and the data is made available for analysis by the community, the I/O performance for iterating over events becomes one of the driving factors in minimizing the “time to insight” for analyses.

Disk space is usually the key concern of the experiment computing models as disk is the most expensive hardware component. The community uses compression extensively to keep the disk costs to a minimum. Investigations in optimizing the storage systems and used representation of data on disk together will be needed, including the utilization of new additional fast storage layers like SSD storage and NVRAM-like storage, which exhibit different characteristics than the currently dominant spinning disk installations. An often stated aim of the baseline analysis model of subsequent data reduction steps is to arrive at a dataset that “can fit on one’s laptop”, while optimizing reusability of the intermediate outputs to minimize “time to insight” for a large number of parallel analyses. The hope is that future hardware infrastructure and new technologies from industry and other science fields might make intermediate steps unnecessary.

Maybe the output of a final “laptop” dataset could be made obsolete using these new approaches and electronic notebooks, which currently are gaining in popularity. It is expected that the late stage analysis will neither be entirely local nor entirely remote. We will need to support both, e.g. support syncing from remote services like notebook based analysis-as-a-service into a local laptop environment to support ‘airplane mode’. Analysis should scale easily from the laptop to cloud/grid resources to special resources. Automatic failure recovery is of great importance, as most time is currently spent on recovering the last 1%.

While the event processing rates and latency to make histograms that one gets on the laptop might be difficult to match when the data is remote, it seems conceivable given future technology evolution, to approach local data performance with elements such as (a) nearby data caching and (b) scheduling of analysis workflows to give time to the system to prepare the data for low-latency interactive access. Further optimization could be gained by switching to a functional or declarative programming model. Instead of having to define and control the “how”, the analyst would declare the “what” of his analysis, essentially removing the need to define the event loop in an analysis and leave it to underlying services and systems to optimally iterate over events. This would open up new possibilities for analysis facilities. R&D in that area will be needed to allow functional analysis programs and new analysis facility concepts to work together seamlessly.

In all cases, reproducibility is becoming something we have to take very seriously. Using a more heterogeneous hardware mix with a more diverse set of techniques and technologies can only complicate the proving of reproducibility of our analyses. Reproducibility needs to be considered in all approaches under investigation and needs to be a fundamental component of the system as a whole.

As analysis is the end of the process, it is nevertheless the place where the majority of physicists interact with the system. It will be a continually evolving paradigm, and as such it is important that facilities and infrastructures are capable of adapting and evolving to support the changing analysis needs.

Summary of R&D Activities

There are suggestions for basic R&D work, and prototyping needed in a number of areas that encompass the range of activities needed to implement the computing models. These can be grouped as follows:

1. Federated data centres (a prototype “Data-Lake”). Tasks include:
    - Understanding the needed functionalities, including policies for managing data and replications, availability, quality of service, service levels, etc.;
    - Understand the limitations and issues;
    - Understand how to interface a data-lake federation with heterogeneous storage systems in different sites (may be in 2nd iteration).
    - Investigate how to define and manage the interconnects, network performance and bandwidth, monitoring, service quality etc. Integration of networking information and testing of advanced networking infrastructure.
    - Investigate policies for managing and serving derived data sets, lifetimes, re-creation (on-demand?), caching of data, etc.
2. Data management and data access.
    - Define a set of data management functionality that is common across experiments, and that is initially well enough understood to implement a prototype.
    - Prototype a storage cache for use at sites that do not manage primary data sets.
    - Prototype a general event server/streaming service as a possible implementation of a Content Delivery Network for HEP, tailored to the HEP data access patterns, data organization and delivery protocols . Should be an extension of the data management tools to support fine-grained data

processing, and streaming for remote sites. This requires a sophisticated bookkeeping service at the required granularity of processing.
   - Investigate the long term replacement of gridftp as the primary data transfer protocol. Define metrics (performance, etc.) for evaluation.
   - Benchmark end-end data delivery for the main use cases (reco, MC, various analysis workloads, etc.), what are the impediments to efficient data delivery to the CPU to and from (remote) storage? What are the necessary storage hierarchies, and how does that map into technologies foreseen?
   - Investigate possible alternatives to the current strategies for data organization, retention, integrity, archival. Explore the possibility to shift boundaries in the storage hierarchy (tape/disk/SSD/..) and consider different policies, evaluating the impact in terms of data access latency, data integrity and cost. Understand how a proposed solution can flexibly adapt to the hardware market price (for example, if the price of tape drives increases beyond expectations). The Community White Paper on Data Organization, Management and Access elaborates more on this concept.
3. Data cataloging and Metadata
   - Investigate the possibility to develop a cataloging service capable to bookkeep the data at the level of granularity needed for data management and data processing. Particularly, in the case of an event service and event streaming service the system needs to be able to collect and address single events or event parcels, at least for the events “in flight”. The possibility to define datasets made of single events collections should also be considered. The system should be capable to store the metadata needed to the data and workload management systems and offer the capability to annotate information about the data itself, again at the required granularity. The Community White Paper on Data Organization, Management and Access elaborates more on this concept.
4. Extension of core facilities.
   - What are the mechanisms by which to easily and transparently plug in resources to the basic data-lake? Examples include provisioning HPC, commercial clouds, and other opportunistic resources.
   - How will those resources be fed with data, and eventually store the results?
   - What are the mechanisms (and policy) for making cost-driven provisioning decisions for commercial use (e.g spot markets etc.). How is scheduling for peak usage done?
5. Technology evolution, performance measurements, benchmarking, cost analysis
   - Track and provide outlook for hardware evolution in all aspects; testing with techlab/openlab style resources
   - How can rack-scale technologies help in resource provisioning?
   - Benchmarking, performance analysis, metrics, etc.
   - Define and develop the “cost model” process for guiding where investments should go.
6. Resource provisioning.
   - HTCondor as a common scheduling layer, either for containers or pilot factories, interconnect to remote resources,

- What is the unit of work at a site - move to containers rather than pilot factories?
- Can we provision task-specific platforms (create and destroy) for specific short term tasks?
- Can we provision and use hardware-specific resources (e.g. non-traditional CPUs, GPUs, etc.) for specific tasks.

7. Workflow and workload management.
    - What does a common layer look like. Can a prototype be implemented based on well-understood functionality?
    - Specify and execute workflow rather than jobs?
8. Analysis. Need to investigate some ideas of how analysis technologies will impact potential facilities, data management, etc.
    - What requirements would ML as a service (or other) have on resources and provisioning.
    - Do cloud-like services provide the necessary direction with potentially scalable backends on demand?

References

[campana] Simone Campana, talk at the ECFA meeting: (https://indico.cern.ch/event/524795/contributions/2236590/attachments/1347419/2032314/ECFA2016.pdf)

[monarc] The MONARC project: http://monarc.web.cern.ch/MONARC

HSF-CWP-001 - *Exploiting HPC for HEP towards Exascale* (link to pdf)

Authors: Wahid Bhimji (LBNL), Taylor Childers (ANL), Lisa Gerhardt (LBNL), Jeff Porter (LBNL)

HSF-CWP-002 - *SDN Next Generation Integrated Architecture (SDN-NGenIA) For HEP and Global Scale Science* (link to pdf); Harvey Newman (Caltech)

HSF-CWP-003 - *Next Generation Exascale Network Integrated Architecture for HEP and Global Science* (link to pdf); Harvey Newman (Caltech), M. Spiropulu (Caltech), J. Balcas (Caltech), D. Kcira (Caltech), I. Legrand (Caltech), A. Mughal (Caltech), J. R. Vlimant (Caltech), R. Voicu (Caltech)

HSF-CWP-004 - *Thoughts about HL-LHC Computing Models* (link to pdf); Maria Girone (CERN), Ian Fisk (Simons Foundation), Oliver Gutsche (FNAL)

HSF-CWP-005 - *Fundamentals of an HL-LHC Computing Model* (link to pdf); Frank Wuerthwein (UCSD)

HSF-CWP-006 - *Evolution to a HL-LHC Computing Data Model* (link to pdf); Elizabeth Sexton-Kennedy (FNAL)

HSF-CWP-007 - *Practical Reproducible Evaluation of Systems with Popper* (link to pdf); Ivo Jimenez (UCSC), Michael Sevilla (UCSC), Noah Watkins (UCSC), Carlos Maltzahn (UCSC)

HSF-CWP-008 - *Ceph: An Open-Source Software-Defined Storage Stack* (link to pdf); Noah Watkins (UCSC), Michael Sevilla (UCSC), Ivo Jimenez (UCSC), Carlos Maltzahn (UCSC)

HSF-CWP-009 - *Evolution of HEP Computing* (link to pdf); Ian Bird (CERN)

HSF-CWP-010 - *Detector Simulation Challenges in Fermilab IF Experiments* (link to pdf); V.D. Elvira (FNAL), L.J. Fields (FNAL), K.L. Genser (FNAL), R.W. Hatcher (FNAL), T.Junk (FNAL), R. Kutschke (FNAL), P. Lebrun (FNAL), M. Mooney (BNL), B. Viren (BNL), …

HSF-CWP-011 - *CMS Simulation in the HL-LHC Era* (link to pdf); S. Banerjee (FNAL), D. Elvira (FNAL), M. Hildreth (FNAL), V. Ivantchenko (Russian Academy of Sciences), K. Pedro (FNAL), I. Osborne (FNAL), S. Sekmen (Kyungpook National University), L. Sexton-Kennedy (FNAL)

HSF-CWP-012 - *ATLAS computing model notes (snapshot 23 Jan 2017)* (link to pdf); S.Campana (CERN), A. Klimentov (BNL), E. Lancon (BNL), T. Wenaus (BNL)

HSF-CWP-013 - *Virtual Data* (link to pdf); Richard Mount (SLAC)

HSF-CWP-014 - *Some notes on Computing Model evolution/tests/expectations for HL-LHC* (link to pdf); Tommaso Boccali (INFN-Pisa)

HSF-CWP-015 - *Deep learning and neutrino physics* (link to pdf); G. N. Perdue (FNAL), T. Golan (University of Wroclaw), A. Himmel (FNAL), E. Niner (FNAL), F. Psihas (Indiana University), and A. Radovic (College of William & Mary)