PROCEEDINGS OF SCIENCE

Integrating Network-Awareness and Network-Management into PhEDEx

Vlad Lăpădătescu*
Caltech / USA
E-mail: vlad@cern.ch

Tony Wildish
Princeton / USA
E-mail: awildish@princeton.edu

ANSE Collaboration †

ANSE (Advanced Network Services for Experiments) is an NSF funded project, which aims to incorporate advanced network-aware tools in the mainstream production workflows of LHC's two largest experiments: ATLAS and CMS. For CMS, this translates in the integration of bandwidth provisioning capabilities in PhEDEx, its data-transfer management tool.

PhEDEx controls the large-scale data-flows on the WAN across the experiment, typically handling 1 PB of data per week, spread over 70 sites. This is only set to increase once LHC resumes operations in 2015.

The goal of ANSE is to improve the overall working efficiency of the experiments, by allowing for more deterministic times to completion for a designated set of data transfers, through the use of end-to-end dynamic virtual circuits with guaranteed bandwidth.

Through our work in ANSE, we have enhanced PhEDEx, allowing it to control a circuit's lifecycle based on its own needs. By checking its current workload and past transfer history on normal links, PhEDEx is now able to make smart use of dynamic circuits, only creating one when it's worth doing so. Different circuit management infrastructures can be used, via a plug-in system, making it highly adaptable.

In this paper, we present the progress made by ANSE with regards to PhEDEx. We show how our system has evolved since the prototype phase we presented last year, and how it is now able to make use of dynamic circuits as a production-quality service. We describe its updated software architecture and how this mechanism can be refactored and used as a stand-alone system in other software domains (like ATLAS' PanDA).

We conclude, by describing the remaining work to be done ANSE (for PhEDEx) and discuss on future directions for continued development.

International Symposium on Grids and Clouds (ISGC) 2015,
15-20 March 2015
Academia Sinica, Taipei, Taiwan

*Speaker.

†B. Ball, A. Barczyk, J. Batista, K. De, S. McKee, A. Melo, H. Newman, A. Petrosyan, P. Sheldon, R. Voicu

PoS(ISGC2015)018

1. Introduction

1.1 PhEDEx

PhEDEx[1] is the data-placement management tool for the CMS[2] experiment at the LHC. It manages the scheduling of all large-scale WAN transfers in CMS, ensuring reliable delivery of the data. It consists of several components:

- an Oracle database, hosted at CERN
- a website and data-service, which users (humans or machine) use to interact with and control PhEDEx
- a set of *central* agents that deal with routing, request-management, bookeeping and other activities. These agents are also hosted at CERN, though they could be run anywhere. The key point is that there is only one set of central agents per PhEDEx instance
- a set of *site-agents*, one set for every site that receives data

PhEDEx was originally conceived over ten years ago now, and the architecture still reflects design decisions made at that time. Then, the network was expected to be the weakest link in the developing Worldwide LHC Grid (WLCG)[3]. Networks were expected to have bandwidth of the order of 100 Mb/sec, to be unreliable, and to be poorly connected across the span of the CMS experiment. Accordingly, PhEDEx will back off fast and retry gently in the face of failed transfers, on the assumption that failures will take time to fix, and that there is other data that can be transferred in the meantime. This can lead to large latencies caused by transient errors, with subsequent delays in processing the data.

The data-transfer topology was designed with a strongly hierarchical structure. The Tier-0 (CERN) transferred data primarily to a set of 6-7 Tier-1 sites, and each Tier-1 site handled traffic between itself, the other Tier-1s. and it's local Tier-2 sites. Tier-2's wishing to exchange data would have to go via Tier-1 intermediaries. This kept the transfer-links (i.e. the set of (source,destination) pairs) mostly in the realm of a single regional network operator, the only cross-region links were used by Tier-1s which were assumed to have the expertise to debug problems and keep the data flowing. It also kept the overall number of transfer links low, since the majority of sites (the Tier-2s) had only one link, to their associated Tier-1 site.

Today, the reality is very different. The network has emerged as the most reliable component of the WLCG; problems with transfers tend to be at the end-points rather than in the network itself. Bandwidths of 10 Gb/sec between Tier-2 sites is common in many areas, and 100 Gb/sec connectivity is starting to appear. Even where the bandwidth is still relatively low, connections are quite reliable, so data can be transferred effectively. This has led CMS to embrace a fully connected transfer mesh in which all sites are allowed to connect to any other site, so the number of transfer links has risen from about 100 to nearer 3000.

1.2 ANSE

CMS has decided to address these limitations, and is considering a number of possible avenues for the future of PhEDEx[4]. The ANSE[1] [5] project is addressing some of them, specifically how to

[1] A project funded by NSF CC-NIE program, initially for two years, which started in January 2013

make PhEDEx more aware of the network status, and how to provide PhEDEx with the means to control the network by way of virtual circuits and bandwidth-on-demand (BoD).

2. Initial prototype

2.1 Introducing circuit awareness

As described, PhEDEx consists of two different types of agents: central and site agents. As such, the integration of dynamic circuits can be done at one of those levels.
Even though at a site level only a local, site centric, view of the network (and transfer queues) exists, it was decided that integrating circuit awareness here, was a good compromise between ideal required functionality and the complexity of the task at hand.
The initial efforts[6] focused on providing a prototype based on the FileDownload site agent. In addition, for the purposes of this prototype, FDT[7] was chosen as the transfer backend. The FDT tool is a fast, lightweight transfer tool, which integrates IDCP[2] [8] OSCARS[9] (On-Demand Secure Circuits and Advance Reservation System) calls.

2.2 Standard FileDownload agent

The FileDownload agent in PhEDEx is one of the site agents[3] and it's responsible for the execution of file transfers. The agent (as all PhEDEx agents) is event driven using the Perl Object Environment framework, operates in pull mode[4] and executes transfers from its transfer queue. The transfer queue is continuously updated by the FileRouter agent and contains information about all due transfers from the different source nodes (site) that the agent has to execute. For each source-destination pair of PhEDEx nodes, the FileDownload agent organizes the files in a way that is suitable for the transfer tool which is going to be used. This consists in splitting the queue of files into separate transfer jobs. Each transfer jobs usually contains several tens of files. The agent executes one or more transfer jobs in parallel, depending on the site's configuration, then verifies that the files have been correctly delivered before reporting back to the database with the transfer results.
At any given time, PhEDEx only knows if two sites have or haven't got connectivity between them, but knows nothing about the physical network path existing between them. This means that when dealing with any pair of source-destination nodes, the FileDownload agent executes the transfer on the default network path available between the two storage servers involved in the transfer. Even if available, the agent can't use alternative transfer paths since it has no knowledge of them.

2.2.1 PFNs and LFNs

PhEDEx transfers files in bulk, in what is called a transfer job. Each transfer job knows the source and destination URLs of the file being transferred as well as monitoring and logging information. In PhEDEx these URLs are called Physical File Names (PFNs) and encode the transfer protocol being used, the hostname (or IP) of the storage on which the file is located and the local path of the

[2]InterDomain Controller Protocol
[3]Each site runs one or more copies of this agent.
[4]Files are downloaded from a source site.

file on the storage. The hostname present in the PFN, doesn't necessarily point to the actual server on which file replicas reside, but points to the entity that knows where the file is located. This is particularly important when dealing with FTS[10] and gridFTP. The PFNs are actually constructed from Logical File Names (LFNs) and a look-up table which each site maintains and uploads to the central database.

2.3 Prototype FileDownload agent

Since the FileDownload agent is instrumental in transferring files between sites, it was picked as the best place where the functionality needed for circuit awareness could be added.

- The FileDownload agent, can now estimate the amount of work remaining to be done for each source-destination pair. This estimation is based on simple monitoring statistics that PhEDEx gathers, and information on its own download queue. If there is significant[5] work remaining to be done, a circuit is requested.
- After a circuit is requested, the agent checks for duplicate circuits, creates a status file (used in case of an abnormal agent shutdown), calls the circuit backend for requesting a circuit, then creates a timeout timer for the reply. If it doesn't receive a reply from the circuit backend in the allotted time, the request is considered as failed.
- Once the reply from the circuit backend is received, the state of the request is updated and saved to disk. A new timer used to teardown the circuit is started.
- The FileDowload agent also routinely verifies that the state of circuits in memory matches what was saved on disk.

2.4 Test setup

The prototype was tested on ANSE's testbed using two PhEDEx test sites (Figure 1). Each test site was composed of a storage server and a PhEDEx site server. On each storage server we used one disk controller managing 8 SSDs. Two 10Gbps virtual circuits were created for the purposes of this test. One of the circuits was used to model a shared link in which PhEDEx had to compete with other traffic. This background traffic was generated by Iperf and consisted of a continuous stream of UDP packets at 5Gbps. The second circuit served as the dedicated link. The main purpose of this test wasn't to show that we can saturate a 10Gbps link with PhEDEX, but that a PhEDEx FileDownload agent is able to switch to using a new path in a transparent manner and with no down time. The first part of the test consisted of a 10 hour run with PhEDEx transfers on the shared link. After this time, PhEDEx switched to using the dedicated circuit and continued transfers for another 10 hours. PhEDEx was setup up to run a single 450 GB transfer job at a time, each one comprised of 30 files of 15 GB each.

2.5 Results

The results are summarised in Figure 2. This plot portrays the transfer speeds that PhEDEx achieved in our test scenario. The left half of the plot shows the PhEDEx throughput in the first part of the test while it was competing with the iperf traffic. Naturally, this limits PhEDEx traffic to 600MB/sec (4800 Gbps). The right part of the plot, displays the transfer speeds PhEDEx achieved

[5] arbitrary limit set initially to 6 hours

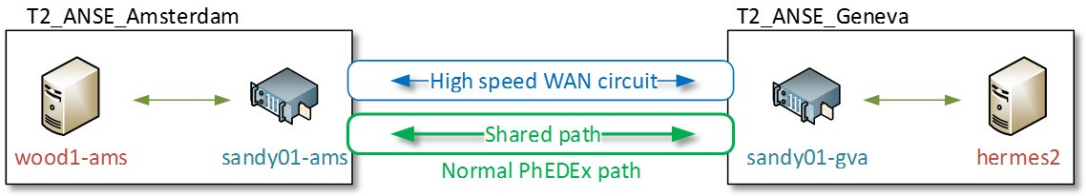

Figure 1: PhEDEx testbed for ANSE

on an empty link, with no competing traffic. An effective doubling of throughput is seen, with PhEDEx being able to fill up the whole 10 Gbps link. The seesaw look of this plot is linked to the fact that PhEDEx has a delay between finishing one job and starting the next. This is due to various factors: pre/post validation, preparation of copyjobs or even time spent by the backend itself before actually launching a transfer. Because of these delays, the average rates reported by PhEDEx (Figure 3) will always be lower than the average rates of each individual transfer job. Lastly, Figure 2 also shows the fact that there was no interruption in service when PhEDEx moved from one link to another. This test demonstrates the potential usefulness of using virtual circuits in PhEDEx.

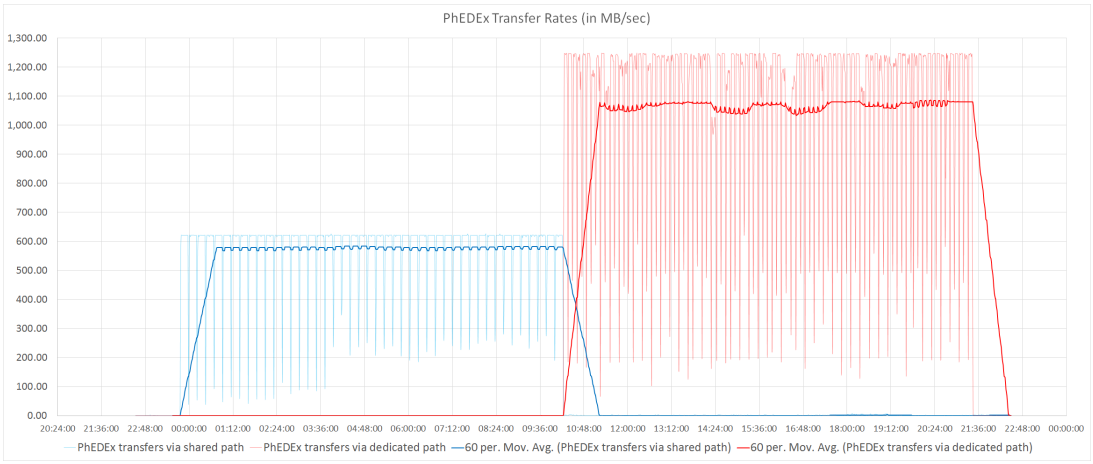

Figure 2: View of PhEDEx-only transfers on both the shared and dedicated path

2.6 Prototype limitations

The prototype proved helpful in demonstrating that circuits can be useful, and that circuit awareness can be integrated into PhEDEx, specifically at the site level. The design decisions taken in the development process mirror the need for a fast working prototype and this ultimately had an impact on its ultimate usefulness.

- Non-modular design: All of the control logic was contained in the FileDownload agent. This meant that the part that requests circuits and manages their lifecycle could not be separated from the FileDownload agent. This was needed in order to create a stand alone application that could be used external tools as well.
- Single circuit backend: Relied on a single method of requesting circuits (DYNES). For every different circuit provider used, a major change in code was required.

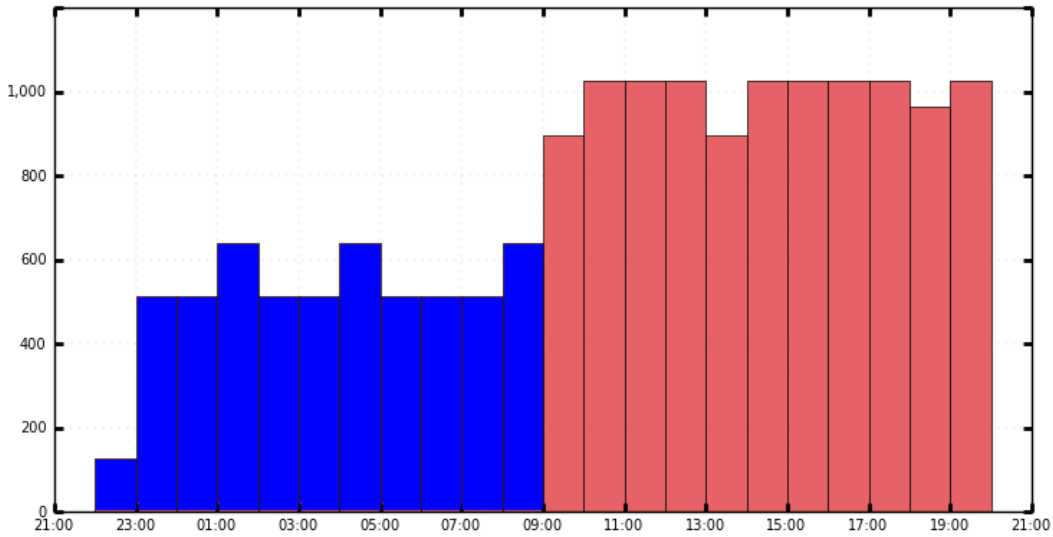

Figure 3: View of PhEDEx only transfers on both the shared and dedicated path

- Relied on FDT as a transfer tool: Unfortunately, this transfer tool is not widely used in production. Our production ready version should at least support the most common protocols out there (ie. FTS/SRM/gridFTP)

In addition to these decisions, it was assumed that a simple storage system would be used. In all transfers involving the prototype, data was always moved from server to server, not from a storage farm to a storage farm. This becomes an issue when when extrapolating from a prototype to a production infrastructure since, it is assumed that the PFNs that PhEDEx receives, correspond to the actual location of files on a server. In this scenario, the endpoints of the circuits are known: the hostnames/IPs in a PFN correspond to a circuit endpoint. In a transfer involving the storage farm, the PFN points to a server which then redirects to one of the replicas available. Until this redirection is done there is no way of knowing which servers will be involved in the transfer, therefore the endpoints of the circuit are unknown.

3. Towards a production ready software

After the success of the prototype we were ready to design the production solution. The production ready version of the system had to:

- Support multiple circuit backends
- Separate all PhEDEx logic from circuit lifecycle management
- Provide a REST interface for external application control
- Provide a robust service to the application, regardless of potential instabilities in the underlying network service

This redesign took the simple extension of the FileDownload agent to the software that is used today, whose class diagram is shown in Figure 4.

3.1 What was changed?

3.1.1 Circuit agent

All of the logic initially inserted in the FileDownload agent has now been extracted and placed in the CircuitAgent, an agent directly extending the FileDownload agent. This opens the possibility of deploying the software in production and only using the circuit integration part with select sites. This is done by just by modifying a line in the PhEDEx config files. Another advantage is that operators can quickly roll back to the normal FileDownload agent if the CircuitAgent misbehaves during integration

The CircuitAgent no longer contains the logic related to the lifecycle management of circuits. This has been delegated to a new entity called the ResourceManager.

3.1.2 ResourceManager

The ResourceManager handles the lifecycle of circuits on behalf of PhEDEx and external programs. It can be viewed as a stand-alone entity, as it can receive calls directly from PhEDEx or interact with external applications (like PanDA[11]) via a new REST API.

The new API supports the following calls:

- createCircuit(to, from, lifetime, bandwidth): creates a circuit
- removeCircuit(to, from): tears down a circuit
- getInfo()
  - RESOURCES: returns all info currently available (circuits active or expired [6])
  - BACKEND_TYPE: returns the circuit backend name used to provision circuits
  - RESOURCE_HISTORY: returns all info on expired circuits
  - LINKS_BLACKLISTED: returns all the links currently blacklisted [7]
  - ONLINE_CIRCUIT, to, from: provides info on a specific active circuit

Another important feature is that the ResourceManager can now use multiple circuit providers via a plug-in system. Two circuit providers are supported at the moment: DYNES[12] and NSI[13]. A third plug-in (Dummy) is provided for test purposes only.

The sequence diagram presented in Figure 5 shows the main events at work in the software, as well as the interaction order between them.

3.2 Circuit providers

When ANSE began it was assumed it would use the DYNES infrastructure to provide Layer 3 circuits between a select number of sites. DYNES was regarded as a "dynamic network cyber-instrument" and spanned over 40 US universities and 14 Internet2 connectors. It was based on the implementation of IDCP (Inter-Domain Circuit Protocol) developed by ESnet and Internet2 (with cooperative development from GEANT and GLIF as well).

As official funding ended for DYNES, the lack of continued support forced a switch to different technologies. To this end, NSI was chosen as the replacement for DYNES as it is presently supported by many important network providers around the globe (ESnet, Internet2, GEANT, etc.)

[6] The queue of expired circuits is limited to the last 1000 entries

[7] If a link is blacklisted, it means no circuit requests can be issued on it until it is whitelisted

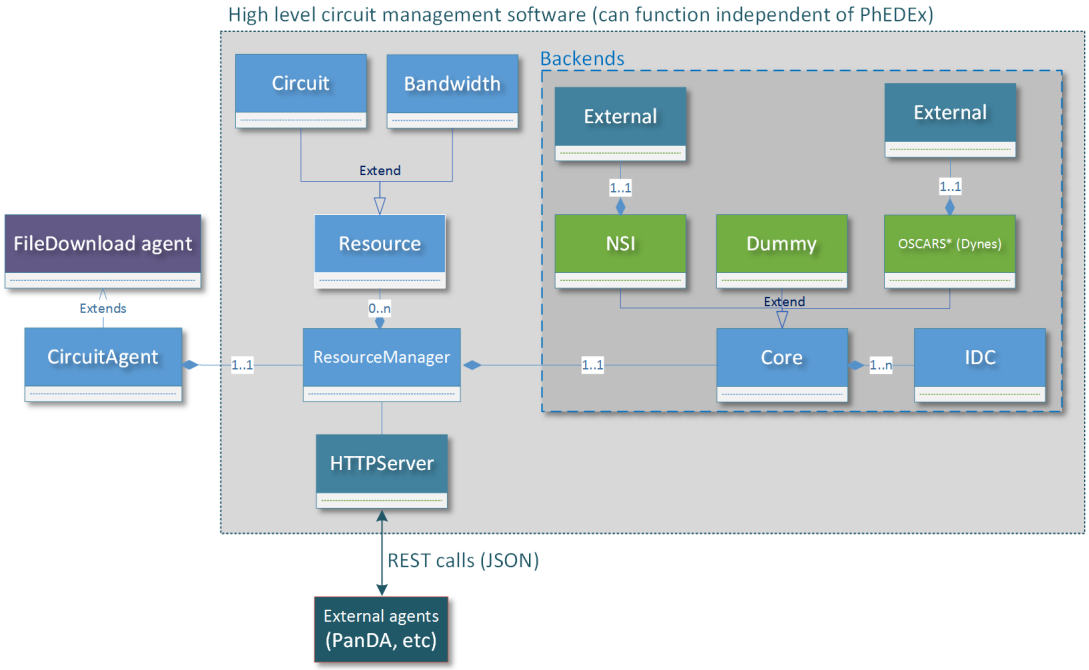

Figure 4: Class diagram of the circuit management software integrated in PhEDEx

3.2.1 What is NSI?

NSI stands for Network Service Interface and is defined as the interface between a NS requestor agent and a NS provider agent, to request a transport connection. The requestor could be a host, middleware or network provider, while the provider could be a home or campus network, or even a national infrastructure provider.
NSI provides different functionalities:

- Resource management: Scheduling, reservation, instantiation, negotiation
- Resource information: Service discovery, topology exchange, monitoring, history, security

NSI[14] supports both tree and chain models of service chaining, and it's effectively a two phase reservation system (Figure 6).

- First phase
  - Reservation is made
  - Availability is checked
  - Resources are (temporarily) held
- Second phase
  - Requestor commits or aborts the reservation
  - Should the requestor fail to act within a certain time limit, the reservation is released

The NSI plug-in implemented in our software uses an NSI CLI tool which was provided by ESnet.

4. Finding a solution

The software presented in the previous section, came a long way from the prototype that preceded

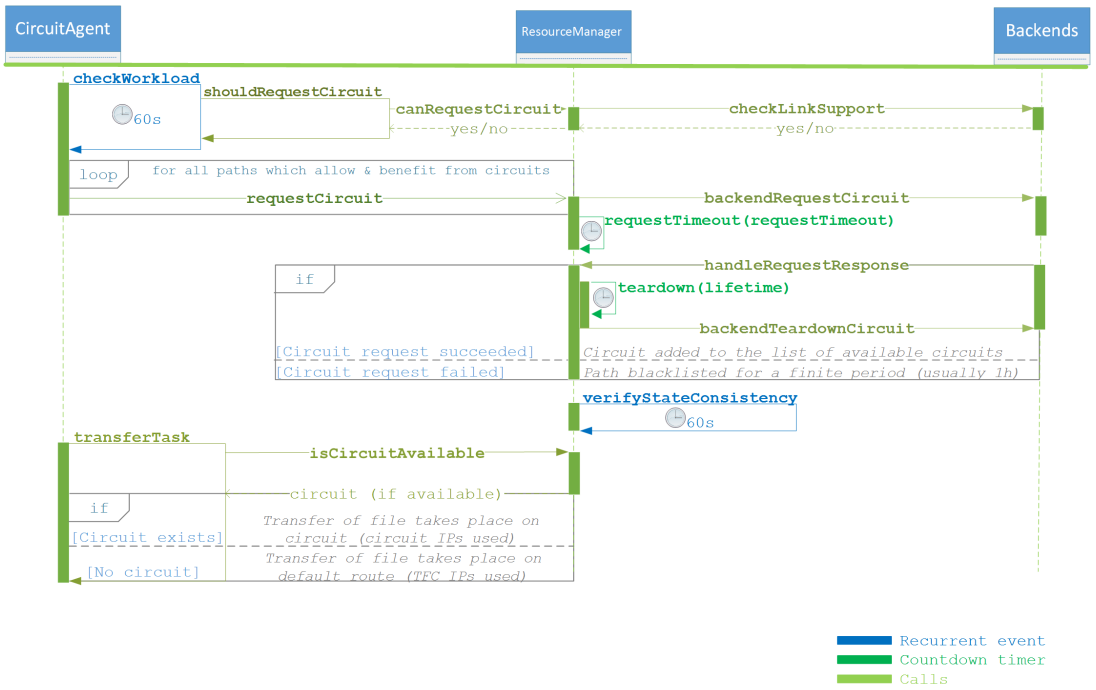

Figure 5: Sequence diagram of the circuit management software integrated in PhEDEx

it, however the complete solution to bringing circuit awareness and management into production is still under development.

4.1 Difficulties in using NSI

Although NSI is now the most popular implementation that promises to provide circuits in a production environment, it comes nonetheless with a few limitations.

Provides a layer 2 circuit: In contrast to Dynes which provided Layer 3 circuits between two servers, NSI only provides a Layer 2 circuit. This means that the transfer backends used in PhEDEx cannot directly use it. A Layer 3 path has to be created on top of the Layer 2 circuit. Constructing such a path means having privileged access to the site's network infrastructure - something that is not trivial, especially with the start of LHC operations in 2015.

Circuit ends at the border router: The Layer 2 path doesn't end at the storage servers, nor does it end at the storage farm's router. The Layer 3 path needs to be created not just on top of the circuit, but also from the border routers to the storage

Guarantee bandwidth is not supported by all providers: The motivation of using circuits in PhEDEx is two fold: provide more deterministic transfer times and provide a way of privileging select traffic. If even a single circuit provider involved in an inter-domain transfer doesn't guarantee the bandwidth which PhEDEx asks for, then the advantages of using circuits in PhEDEx may be lost.

NSI adoption is still limited: Even though NSI has a lot more support from big network providers, adoption into production is still limited at this time.

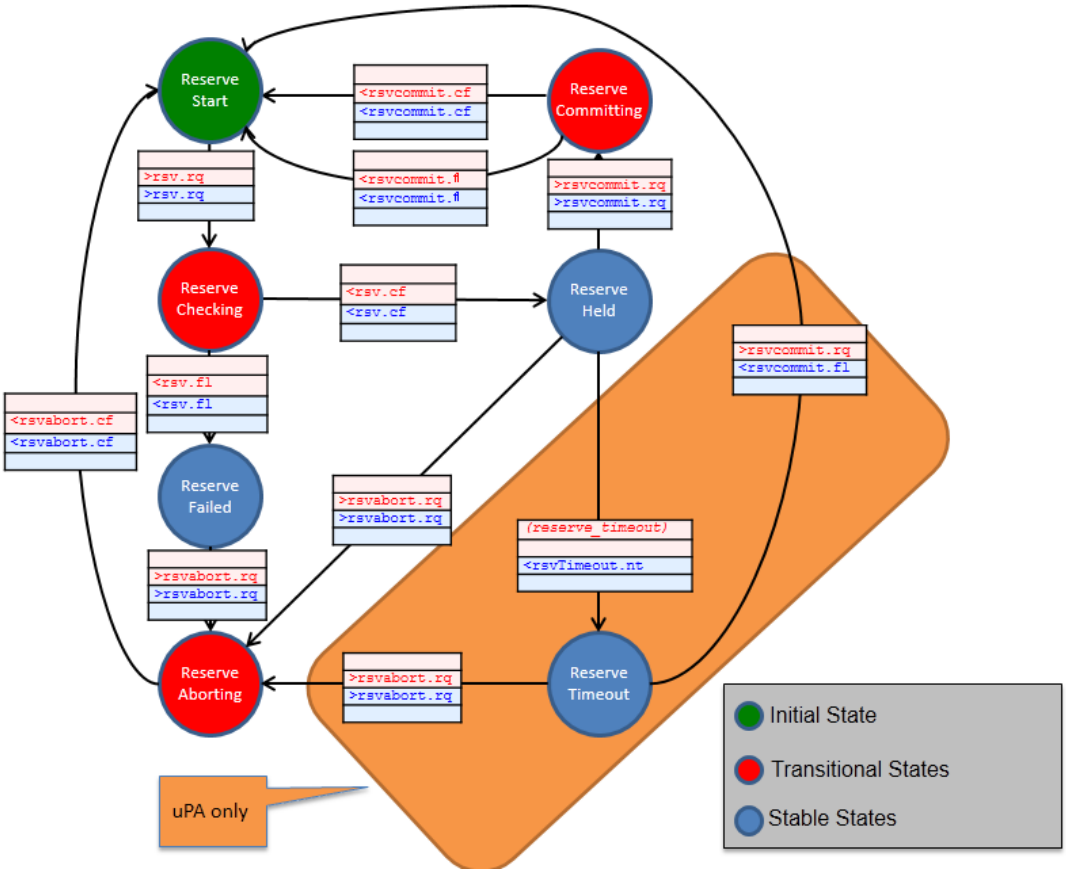

Figure 6: The NSI Reservation State Machine

4.1.1 Difficulties in dealing with Layer2 circuits

Since the transfer backends can't directly used Layer 2 circuits, a Layer 3 path needs to be established between the storage servers, or at least between the routers of the storage farms.
Establishing a Layer 3 path, is non-trivial since:
It requires topology and routing info: PhEDEx is a very high-level software. It only knows about its transfer queue, name of sites, and name and sizes of datasets, blocks and files. Its monitoring information is limited and the PFNs involved in the transfer don't point directly to the file replica used in the data movement.
Direct access to the site's network: Perhaps the biggest challenge in establishing a Layer 3 path is the fact that one requires direct access to the site's network infrastructure. Doing live modifications of the routing information on any production network is controversial enough, even more so given that the infrastructure is critical to the success of LHC's Run 2.

4.2 Difficulties in using FTS/SRM

Not all difficulties in coming up with a solution stem from using NSI as a circuit backend. FTS/SRM and gridFTP are the most popular protocols involved in data transfer and naturally any solution that we propose should at the very least work with them.

The problem here is best described in Figure 7.
In a (PhEDEx) file transfer, two PFNs are always involved: the source and destination PFNs. Ideally, these PFNs should directly point to the physical storage location of the files that need to be transferred. With FTS and SRM, the PFNs don't reflect that. They are only Storage URLs (SURL) and they point to the storage farm's FTS server. In order to get the exact location of the file, FTS needs to pick a file replica from one of the storage elements. Only once that's done, and the SURL is transformed into a Transfer URL (TURL), can a transfer begin.
This is particularly important because the actual servers involved in the transfer, must be known in order to establish circuits between them.

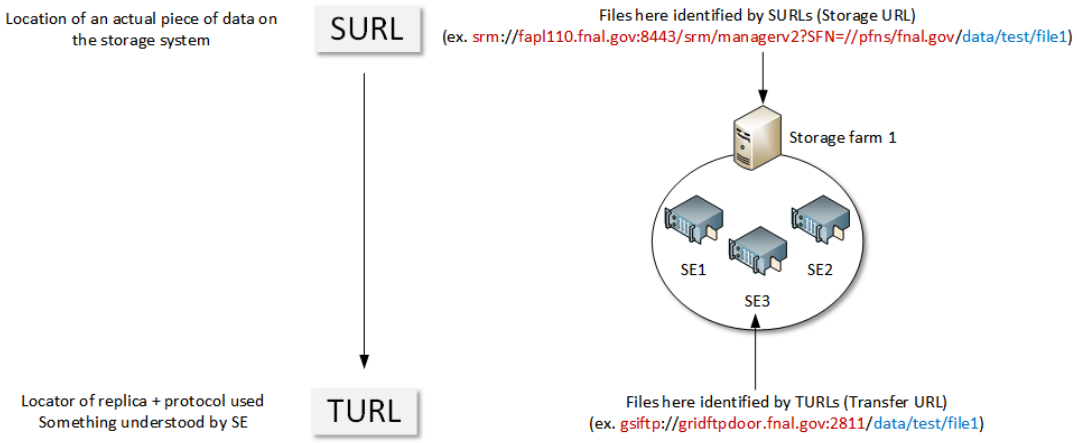

Figure 7: View of PhEDEx-only transfers on both the shared and dedicated path

4.3 Proposed solution

Ideally, any proposed solution should:

- Work in a multiple VO environment
- Try and deal with privileged and unprivileged traffic on the same path
- Work with FTS/SRM and gridFTP
- Be as un-intrusive into the site's operations as possible

The solution which was put forward is presented in Figures 8 and 9.

- The ResourceManager is used to request a Layer 2 circuit between the border routers of sites A and B. The request can come from either PhEDEx or any external application that uses our REST API.
- Once the Layer 2 circuit is up:
  - A wrapper is used to retrieve all the storage servers (looking in the TURLs) involved in the transfer. This information is passed to an OpenFlow controller
  - The ResourceManager informs the OpenFlow controller that a new Layer 2 path is available.
- The OpenFlow controller adds routing information in all the OpenFlow switches, directing all traffic coming from servers involved in the transfer, onto the newly created circuit

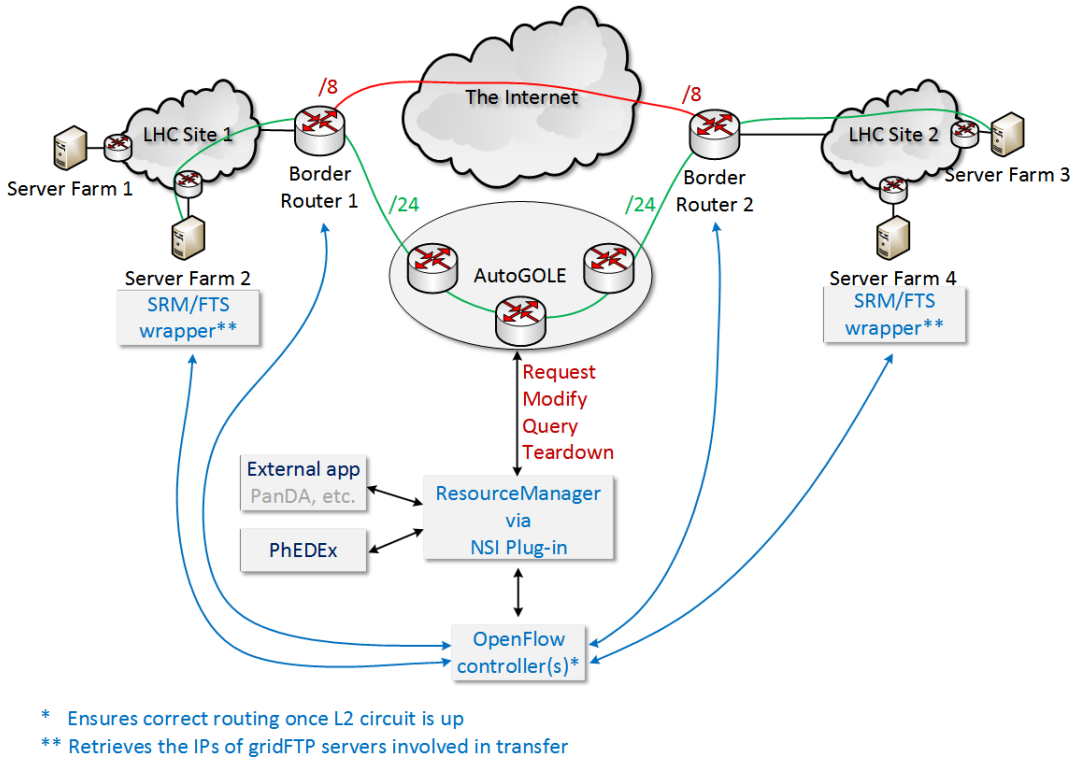

Figure 8: Global view of our proposed solution

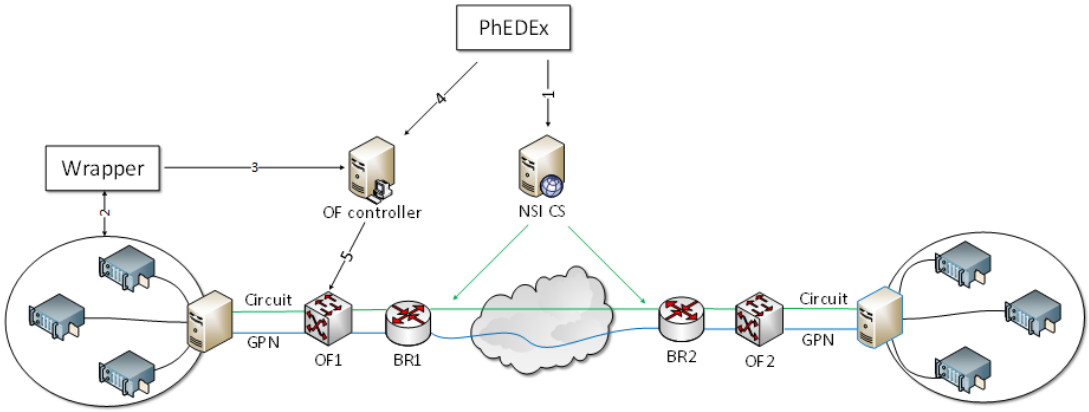

Figure 9: In depth view of our proposed solution

5. Summary and future plans

PhEDEx has been very successful at managing data-flows on the WAN for the CMS collaboration. Nonetheless, its architecture is based on design decisions that start to become invalid, and in order to continue to scale and perform for the future, it must evolve, taking advantage of new technologies.

Over the course of the past two years, the ANSE project has made significant progress towards integrating network awareness into PhEDEx. A prototype was created and tested , demonstrating

the essential features required for circuit integration into PhEDEx. Using the lessons learned during the prototype development, a new improved version was designed, which is going to be released in production shortly. The new system is capable of functioning independently from PhEDEx, it can be controlled remotely via a REST interface and it is modular, being able to use multiple circuit providers.
The remaining work in ANSE, focuses more on the network aspect of the solution. While the PhEDEx software is ready to make use of circuits as soon as they mature into a production ready version, the interim solution still relies on more work on our part, mainly in extending a circuit from storage to storage. This is where ANSE will invest its resources in the following months.

References

[1] Egeland R, Metson S and Wildish T 2008 Data transfer infrastructure for CMS data taking, *XII Advanced Computing and Analysis Techniques in Physics Research (Erice, Italy: Proceedings of Science)*

[2] The CMS Collaboration 2008 The CMS experiment at the CERN LHC *JINST* **3** *S08004*

[3] Eck C *et al.* 2005 LHC Computing Grid Technical Design Report *CERN-LHCC-2005-024*

[4] Bonacorsi D and Wildish T 2013 Challenging CMS Computing with Network-Aware Systems *submitted to CHEP 2013*

[5] LHCONE Point-to-Point Service Workshop, December 2012 *http://indico.cern.ch/event/215393/session/1/contribution/8/material/slides/1.pdf*

[6] International Symposium on Grids and Clouds 2014 *http://pos.sissa.it/archive/conferences/210/021/ISGC2014_021.pdf*

[7] Fast Data Transfer (FDT) *http://fdt.cern.ch/*

[8] InterDomain Controller Protocol *http://www.controlplane.net/*

[9] Guok C, Robertson D, Thompson M, Lee J, Tierney B, Johnston W 2006: Intra and Interdomain Circuit Provisioning Using the OSCARS Reservation System *ICBCNS 2006*

[10] Alvarez Ayllon A, Kamil Simon M, Keeble O and Salichos M 2013 FTS3 - Robust, simplified and high-performance data movement service for WLCG *submitted to CHEP 2013*

[11] ATLAS collaboration, 2008: PanDA: distributed production and distributed analysis system for ATLAS *Journal of Physics: Conference Series, Volume 119, Part 6*

[12] Dynes: DYnamic NEtwork System *https://www.terena.org/activities/e2e/ws3/slides/101129-dynes-Artur.pdf*

[13] OGF Network Service Interface *https://www.terena.org/activities/e2e/ws2/slides2/11_NSI_Eduard.pdf*

[14] NSI Connection Service v2 *https://www.ogf.org/documents/GFD.212.pdf*