Application Aware Software Defined Flows of Workflow Ensembles

George Papadimitriou[†], Eric Lyons[‡], Cong Wang[*], Komal Thareja[*], Ryan Tanaka[†],
Paul Ruth[*], J. J. Villalobos[§], Ivan Rodero[§]
Ewa Deelman[†], Michael Zink[‡], Anirban Mandal[*]

[*]RENCI, University of North Carolina at Chapel Hill, NC, USA
[†]Information Sciences Institute, University of Southern California, CA, USA
[‡]Electrical and Computer Engineering Department, University of Massachusetts at Amherst, MA, USA
[§]Rutgers Discovery Informatics Institute, NJ, USA

***Abstract*—Computational science depends on complex, data intensive applications operating on datasets from a variety of scientific instruments. A major challenge is the integration of data into the scientist's workflow. Recent advances in dynamic, networked cloud resources provide the building blocks to construct reconfiguration, end-to-end infrastructure that can increase scientific productivity, but applications are not taking advantage of them. In our previous work, we introduced DyNamo, that enabled CASA scientists to improve the efficiency of their operations and effortlessly leverage capabilities of the cloud resources available to them that previously remained underutilized. However, the provided workflow automation did not satisfy all the operational requirements of CASA. Custom scripts were still in production to manage workflow triggering, while multiple layer2 connections would have to be allocated to maintain network QoS requirements. In this work, we enhance the DyNamo system with ensemble workflow management capabilities, end-to-end infrastructure monitoring, as well as more advanced network manipulation mechanisms. To accommodate CASA's operational needs we also extended the newly integrated Pegasus Ensemble Manager with file and time based triggering functionality, that improves managing workflow ensembles. Additionally, Virtual Software Defined Exchange (vSDX) capabilities have been extended, enabling link adaptation, flow prioritization and traffic control between endpoints. We evaluate the effects of the DyNamo's vSDX policies by using two CASA workflow ensembles competing for network resources, and we show that traffic shaping of the ensembles can lead to a fairer use of the network links.**

***Index Terms*—adaptive weather sensing, network-centric platform, distributed cloud infrastructure, dynamic network and resource provisioning, malleable data flows, scientific workflow automation, virtual software defined exchange**

I. Introduction

Computational sciences depend on many complex, data-intensive applications to coordinate computations on distributed datasets originating from a variety of scientific instruments and repositories. A major challenge for these applications is to effectively move the data among the diverse compute and storage sites, as well as to integrate the data into the scientists' workflows. These workflows may require specialized access to resources as well as a significant amount of data transfers between tasks, with data residing in different domains. Such scientific workflows require an integration of two or more existing infrastructures using high-performance networks and data management software in order to increase the rate of scientific output. Currently, such integration is either not available, or is purposely-built manually for a specific scientific application or community. However, recent advances in dynamic networked cloud infrastructure, such as ExoGENI [1], provide the technical building blocks to construct and manage such integrated, reconfigurable, end-to-end infrastructure, built-to-order with isolated resources that satisfy workflow compute and data movement requirements.

Data-driven applications and workflows have not adequately taken advantage of the rich set of capabilities offered by a new set of dynamic, networked infrastructures. They are not designed to utilize adaptive features offered by state-of-the-art, networked cloud infrastructures, especially with respect to managing end-to-end, high-performance data flows. As a result, domain scientists in weather modeling, ocean sciences, seismology, etc., struggle to analyze data available in community resources. They often download the data to their own environment, processing it at limited scales in modest chunks, losing crucial time to react to the observed phenomenon and/or missing longitudinal patterns.

Additionally, managing the execution of workflow ensembles over the sophisticated inter-domain infrastructures remains a significant challenge. Traditional workflow management approaches make use of statically provisioned, dedicated, pre-configured compute and network infrastructure. Such approaches are often associated with high cost, since the resources are usually provisioned such that the highest workload can be handled. This imposes extra cost when the system stays idle. Therefore, the bursty computational and network demands for science workflows warrant flexible processing solutions on diverse infrastructures for computing, and malleable, high-performance data movements for efficient data delivery.

Previously, we presented the DyNamo system [2] that addresses the above challenges and we focused on its networking capabilities that enable high-performance, adaptive, performance-isolated data-flows across a federation of distributed cloud resources and community data repositories.

In this paper we enhance the DyNamo system with more advanced layer 2 network manipulation We integrate DyNamo with the virtual Software Defined Exchange (vSDX) [3] architecture, which serves as a virtual interconnect among different domain infrastructures providing flexible, high-performance data transfer over dedicated network circuits. We also introduce end-to-end infrastructure monitoring and new workflow ensemble management capabilities. Specifically, in this paper we make the following contributions:

- We present a data-driven science application, named Collaborative Adaptive Sensing of the Atmosphere (CASA), and describe its revised requirements and challenges that need to be addressed by the DyNamo system.
- We briefly present the architectural components of the DyNamo system, which provides federated infrastructure support to enable malleable, high-performance data flows between diverse, distributed, national-scale research cloud platforms (ExoGENI [1] and Chameleon [4]) and the CASA data repository.
- We present the architecture of the vSDX network infrastructure, which enables high-performance data transfer among heterogeneous compute and storage infrastructures, and we describe the newly introduced functionality that allows link adaptation, flow prioritization and traffic shaping.
- We improve the workflow automation features of the Pegasus workflow management system by enhancing the Pegasus Ensemble Manager with file and time workflow triggering functionality, that can be leveraged directly by CASA workflows to parallelize and control workflow ensembles.
- We provide an in-depth evaluation and analysis of the network performance on the inter-domain, multi-cloud infrastructures. While network resource sharing is unavoidable, we discuss how DyNamo can mitigate the negative effects.

The rest of this paper is organized as follows: Section II discusses background information for the DyNamo components. Section III introduces the components of the CASA weather forecasting application. Section IV presents the extended components that work together to support science workflows. In Section V, we evaluate the performance of the DyNamo ensemble manager and of the DyNamo networking performance. Section VI presents related work. Finally, Section VII concludes the paper.

II. Background

A. Pegasus WMS

Pegasus [5] is a popular workflow management system that enables users to design workflows at a high-level of abstraction. The Pegasus workflow descriptions are independent of the resources available to execute the workflow tasks and are also independent of the location of data and executables. Pegasus transforms these abstract workflows into executable workflows that can be deployed onto distributed and high-performance computing resources such as Leadership Computing Facilities (e.g., NERSC [6] and OLCF [7]), shared computing resources (e.g., XSEDE [8], OSG [9]), local clusters, and commercial and academic clouds (e.g., ExoGENI [1], Chameleon [10]). During the compilation process, Pegasus performs data discovery, locating input data files and executables. Data transfer tasks are automatically added to the executable workflow and perform two key functions: (1) stage in input files to staging areas associated with the target computing resources, and (2) transfer the generated outputs back to a user-specified location. Additionally, data cleanup (removal of data that is no longer required by the workflow at the execution site) and data registration tasks (that catalog the output files) are also added to the workflow. To manage user's data, Pegasus interfaces with a wide variety of backend storage systems that use different data access and transfer protocols.

Pegasus relies on HTCondor [11] DAGMan as its workflow execution engine to run and manage the generated executable workflows. DAGMan in turn, submits the workflow jobs, as they become ready to run (when all parent jobs have completed successfully) to the internal job queue managed by HTCondor. During workflow execution, provenance information from workflow and job logs is automatically parsed and stored in a relational datastore by a monitoring daemon [12].

B. HTCondor

HTCondor [11] is a comprehensive job management system. In contrast to other batch systems such as PBS [13] and SLURM [14], it is particularly suited for distributed high throughout computing (HTC) environments, where one can setup a compute pool of nodes connected over a local area network or a wide area network. HTCondor provides users with a local job queue managed by a daemon *HTCondor Schedd* to which users submit jobs. Furthermore, HTCondor supports matchmaking [15] that allows users to match their jobs with compute nodes that support specific resources. The matchmaking takes place during the negotiation of the resources and is based on HTCondor ClassAds advertised by the compute nodes. Finally, in addition to submitting jobs to HTCondor managed compute resources, HTCondor also provides a component, called HTCondor-G [16], that allows users to submit jobs to other types of schedulers.

C. Mobius

A network-centric platform called Mobius [17] depicted in (Figure 1) includes (a) support for integrated, multi-cloud resource provisioning and for high-performance science data flows across diverse infrastructures, and (b) enhanced mechanisms for interacting with higher level application and workflow management systems and transforming high-level resource requests to low-level provisioning actions, thereby bridging the abstraction gap between data-driven science applications and resource provisioning systems, and (c) transparently maintain the quality of service of the provisioned end-to-end infrastructure through continuous monitoring and control. Mobius was enhanced in our previous work [2] to support

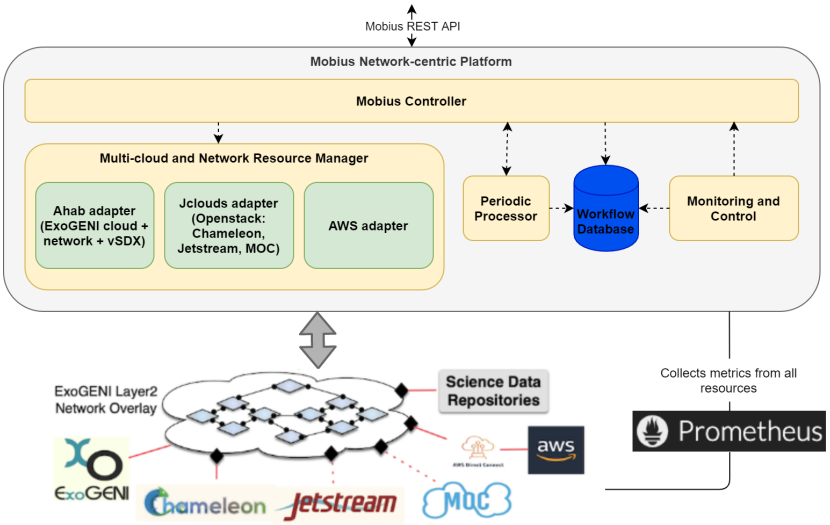

Fig. 1. Mobius - Network centric Platform.

the provisioning of network connections between compute resources across sites/clouds and modulating the bandwidth on these network connections.

D. DyNamo

Data-driven workflows need to automatically and flexibly provision resources to satisfy scientists' bursty computational and network demands. In the case of CASA workflows (Section III), the nature of ever-changing weather events, the number of available sensors, and end user-defined triggers all contribute to load variability.

As presented in a previous work [2], DyNamo enables CASA scientists to transparently acquire cloud resources from multiple cloud providers based on high-level resource requirements. As depicted in Figure 2, DyNamo provides network integration and programmatic provisioning of specific cloud resources using their native APIs. With this approach, domain scientists no longer need to directly interact with diverse cloud providers. To achieve this goal, DyNamo brings together the 3 major components as defined earlier in this section: *Pegasus WMS* is used to provide workflow automation to the applications. *HTCondor* is used to manage the computational resources and distribute the computations. *Mobius* is used to allocate compute and network resources and create the interconnect between data sources and execution sites. Later in Section IV, we will present additional components for DyNamo, making it an integrated, network-aware instrument and monitoring tool for data-driven science applications in multi-cloud environments.

E. Target Cyberinfrastructure

In this paper, we make use of two national scale research cloud providers: ExoGENI and the Chameleon cloud.

- **ExoGENI** [1] is a networked Infrastructure-as-a-Service (IaaS) testbed that links 20 cloud sites on campuses across the US through regional and national transit networks, such as Internet2 [18] and ESnet [19]. ExoGENI allows users to dynamically provision isolated "slices" of compute and networking resources from multiple sites and to integrate various resources using layer2 global dynamic-circuit networks like Internet2 and ESnet, and private clouds like OpenStack [20]. ExoGENI allows users to instantiate customized, distributed topologies, and by provisioning the appropriate network resources corresponding to the topologies, thereby creating end-to-end layer-2 paths.
- **NSF Chameleon Cloud** [10] is a large-scale, deeply programmable testbed designed for systems and networking experiments. Similar to ExoGENI, it leverages OpenStack to deploy isolated slices of cloud resources for user experiments. However, ExoGENI scales in geographic distribution, while Chameleon scales by providing large amounts of compute, storage, and networking resources spread across two sites: University of Chicago (UC) and the Texas Advanced Computing Center (TACC). Chameleon provides over 15K cores and 5 PB storage across the two sites. Users can provision bare metal compute nodes with custom system configuration connected to user-controlled OpenFlow switches operating at up to 100 Gbps. In addition, Chameleon networks can be stitched to external partners including ExoGENI slices.

III. CASA - Motivation

The NSF Engineering Research Center for Collaborative Adaptive Sensing of the Atmosphere (CASA) was formed to study the lower atmosphere with networks of high resolution Doppler weather radars with the goal to improve severe weather awareness [21]. The volumetric data produced by these continuously operating remote sensors must be distributed to processing servers quickly and efficiently such that analysis can occur in near real time for the sake of warning the public to fast developing threats such as tornadoes

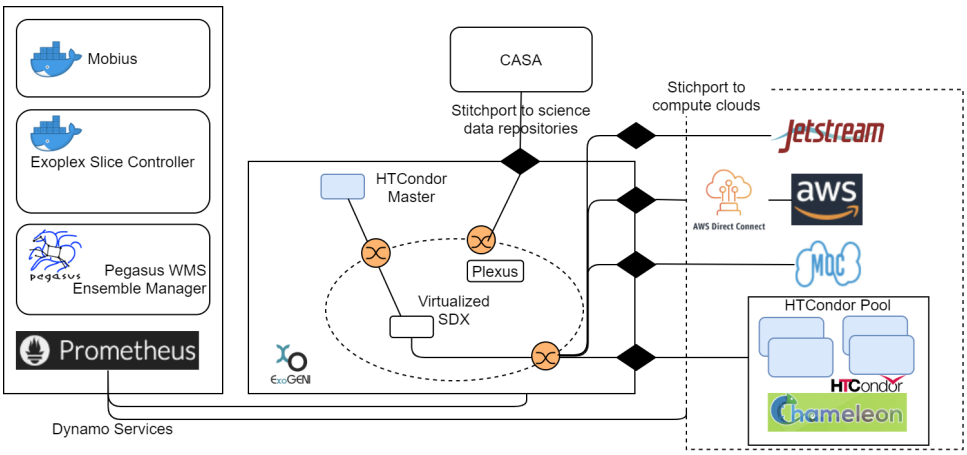

Fig. 2. Dynamo framework

and high winds. The networked radar concept requires that asynchronous raw data from multiple sources are blended together to create value-added meteorological products. At any given time the characteristics of the ongoing weather regime determine the necessity and priority of certain products. For example, a hail detection algorithm takes on high importance only when strong thunderstorms are ongoing, whereas forecasting algorithms may be of more importance well in advance of such severe weather events and perhaps somewhat less so once the event has started.

For years, CASA's scientific workflows associated with product creation have been executed on dedicated servers existing at individual radar sites and at compute centers at NOAA Southern Region Headquarters and at the University of MA Amherst. Servers have been assigned dedicated processing tasks carefully tailored to their hardware and networking resources through trial and error with estimates made regarding the largest likely compute loads associated with each task. The careful management required implies that reconfiguration is highly complex and not feasible by an operator on short notice during an event. To help mitigate this limitation, and to create a more scalable system, in recent years CASA has developed several containerized scientific workflows for calculating these weather products that can be deployed and prioritized as needed [2]. CASA workflows are generally multi-step processes that can include a collection of necessary radar and non-radar sensor data access, grid transformations, format conversions, derived product creation, raster image generation, contouring, GIS based data extraction, and customized notification and alerting [22]. These require complex scheduling and in some cases significant resource consumption, especially during widespread impactful weather when they take on their greatest utility to the end users. For these workflows, CASA now relies on Mobius to provision and modulate compute and networking resources on demand, and uses the Pegasus Workflow Management System to manage task scheduling [2].

We briefly introduce the weather products that are generated by the CASA workflows described in this paper.

A. Nowcast

Nowcasts are short-term advection forecasts that use observed reflectivity data from multiple radars, composite them for a certain number of minutes, and project into the future by estimating the derivatives of motion and intensity with respect to time [23], [24]. Every minute the CASA nowcasting system generates 31 grids of predicted reflectivity, one for each minute into the future from minutes 0-30. The workflow associated with Nowcasting creates raster images for all 31 grids every minute, and also contours for multiple reflectivity levels on each of these grids. The contours are sent to a database where they are used for notification purposes as simplified boundaries containing forecast reflectivity levels of importance for particular applications such as route planning, deployment of spotters, and keeping emergency responders out of harm's way. Nowcast rasters and contours are sent to CASA's data repository over layer 2 stitchports [1] where they are used in web and mobile applications.

B. Wind Speed

A Doppler radar is able to estimate the velocity of moving objects based on a phase shift that occurs if the objects are moving toward or away from the radar beam. Components of velocity perpendicular to the beam are not sensed. For a given radar this means that there will be substantial underestimations of true wind speed over portions of the sensing domain where certain directional components of the winds are not able to be sampled. However, with an overlapping network of radars (as in CASA's case), areas not adequately sampled by one radar are often better sampled by other radars with different relative angles. Therefore CASA's maximum observed velocity workflow ingests the single radar base data from all of the radars in the network and creates a gridded product representing the maximum observed wind speeds. As part of this workflow, areas of severe winds are identified, contoured, and checked against the location of known infrastructure, with email alerts sent out to locations likely to be affected. Workflows that use the large single radar raw data as input have a substantially higher network bandwidth requirement than those operating on

derived data. Input rates of over 100Mbps are common, and given that high winds, which are associated with tornadoes and downbursts are often short lived, one must minimize transmission delays as much as possible to adequately provide warnings for users downstream of the observations.

IV. Approach - DyNamo Extensions

In order to accommodate different application QoS policies and make a more efficient use of the infrastructure, we are extending the DyNamo system (Figure 2) with a more advanced network manipulation component and support for interval based workflow triggering.

A. vSDX module

A Virtual Software Defined Exchange (vSDX) is defined as a virtual interconnect point between multiple adjacent domains, e.g, instruments, compute resources, or data/storage systems. Like a static SDX, a vSDX uses Software Define Networking (SDN) within the exchange to enforce different network policies.

In our case, the vSDX support is provided by the ExoPlex [25] network architecture depicted in (Figure 3). ExoPlex uses an elastic slice controller to coordinate dynamic circuits and the Zeek (formerly Bro) [26] security monitors via Ahab [27]. The controller runs outside of the vSDX slice and exposes a REST API for clients to request network stitching and connectivity and to express QoS parameters. Clients (i.e. Mobius) invoke this API to bind named subnets under its control to the vSDX via L2 stitching and request bandwidth provisioned connectivity with other subnets. The vSDX slice is comprised by virtual compute nodes running OpenVSwitch [28], OpenFlow controllers [29], and Zeek traffic monitors. Traffic flow and routing within the vSDX slice are governed of a variant of the Ryu [30] rest router [31] SDN controller. The vSDX slice controller computes routes internally for traffic transiting through the vSDX network, and invokes the SDN controller API to install them. The SDN controller runs another Ryu module (rest ofctl) to block traffic from offending senders. If a Zeek node detects that traffic violates a Zeek policy, it blocks the sender's traffic by invoking a rest ofctl API call via the Zeek NetControl plugin.

As client requests for bandwidth provisioned connectivity arrive at the vSDX, the slice controller instantiates slice resources as needed to carry the expected traffic. These resources include peering stitchport interfaces at each point of presence (PoP), the OVS nodes that host these vSDX edge interfaces, Zeek (Bro) nodes to monitor the traffic, and backplane links to carry the traffic among the PoPs. The controller reuses existing resources in the slice if they have sufficient idle capacity to carry the newly provisioned traffic, and instantiates new resources as needed. In particular, it adapts the vSDX backplane topology by allocating and releasing dynamic network circuits as needed to meet its bandwidth assurances to its customers. The flows are inspected by out of band Zeek network security monitor appliances to detect intrusion. As a simple form of intrusion prevention, it uses Zeek's NetControl framework to interrupt all traffic from the source of a suspect flow. The vSDX controller deploys Zeek instances elastically to scale capacity.

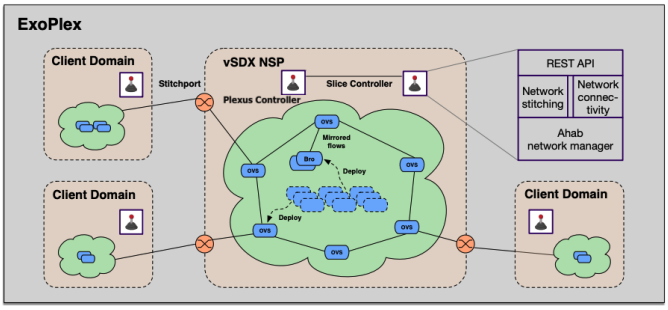

Fig. 3. Virtual Software Defined Exchange (SDX) Network Architecture

In our scenario, the Exoplex Slice controller [32] runs as a docker container. Mobius has been enhanced to communicate with the ExoPlex Slice controller via its REST API to establish network connectivity between ExoGENI and Chameleon via layer2 networks and to allocate bandwidth to individual workflows. Once connectivity is established, Mobius triggers REST API calls to publish network prefixes, sets up routes between network prefixes and dynamically applies different bandwidths as needed. Additionally, we have implemented a Python based interface that can be used to provision the required resources. This interface enables programmatic resource provisioning and is capable of spinning up resources, establishing connectivity and implementing network QoS policies on a per workflow ensemble level.

B. Pegasus Ensemble Manager

Pegasus can manage collections of related workflows, referred to as ensembles, through a service called the Pegasus Ensemble Manager (Pegasus-EM) [33]. Pegasus-EM supports ensemble creation, workflow prioritization, throttling of concurrent executions, and ensemble level monitoring capabilities.

To support dynamic execution of workflow ensembles based on the continuous flow of data obtained from various sources, we have have extended Pegasus-EM with a file based workflow triggering capability. For a given ensemble, Pegasus-EM can create a trigger that watches for newly received files, using a configurable time interval and a given file pattern. On each interval, a Pegasus-EM trigger identifies any new input files that match the provided file pattern, and pass them to a user-defined workflow generation script to dynamically create and plan a Pegasus workflow based on the incoming data. Pegasus-EM executes the workflow generation script and queues up the generated workflow for execution.

C. Prometheus Monitoring

The Prometheus monitoring system [34] has been added to the DyNamo ecosystem. Mobius automatically configures the Prometheus node exporter [35] on each compute node to push system metrics to a Prometheus server hosted at RENCI. The metrics collected by Prometheus enable us to dynamically

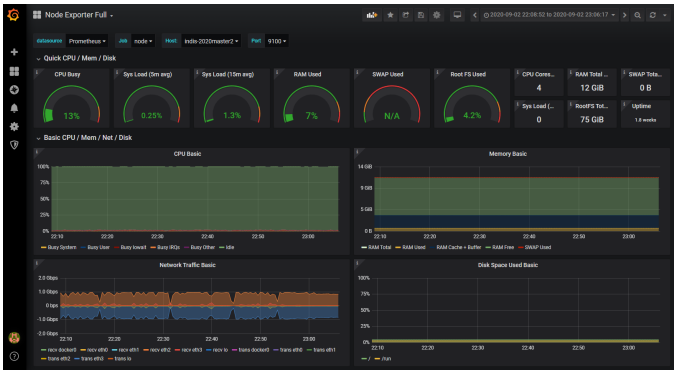

Fig. 4. Grafana Dashboard depicting Prometheus Metrics

take actions to ensure the infrastructure QoS. The actions include enabling compute, storage and network elasticity, i.e., growing and shrinking compute or storage resource pools and increasing or decreasing network properties of links. To visualize the collected data in a comprehensive and easy to understand way, an instance of Grafana [36] has been configured to pull the metric data from Prometheus and plot various graphs on a dashboard depicted in Figure 4.

D. Operational Effect on CASA's Workflows

CASA workflows, due to their nature, can benefit from both of these enhancements to the DyNamo framework. As described in Section III, CASA workflows need to process and respond to a continuous flow of weather radar data arriving at different rates. With the additions to the Pegasus-EM, CASA workflows can be started automatically as new files arrive at CASA's data repository, with direct support by the DyNamo framework. In the past this functionality was implemented using perl scripts that were invoked manually at the processing initiation stage. Moreover, with the introduction of the vSDX capabilities CASA workflow ensembles can now share the same layer2 link in an isolated fashion. I.e, traffic from one workflow can only consume the maximum assigned bandwidth without impacting the network resources assigned to other workflows. CASA's workflows have different requirements among them that not only depend on the data being processed and the pipeline, but also the workflow configuration. With the vSDX, CASA can reserve a single layer2 circuit to its data repository while distributing the network bandwidth based on the network subnet each worker node resides in. Each worker is assigned a specific CASA workflow ensemble by advertising a target workflow tag in its HTCondor advertisements. Previously this functionality was supported by reserving multiple layer2 circuits on CASA's data repository, but due to the limited number of the available links this couldn't be achieved consistently.

V. Evaluation

A. CASA Pegasus Workflows Description

For the evaluation of the QoS impact we have selected two CASA workflows that produce nowcasts and wind speed estimates as described in Section III. The workflow tasks include input data collection and product generation, visualization, contouring into polygon objects, spatial comparisons of identified weather features with infrastructure, and dissemination of notifications.

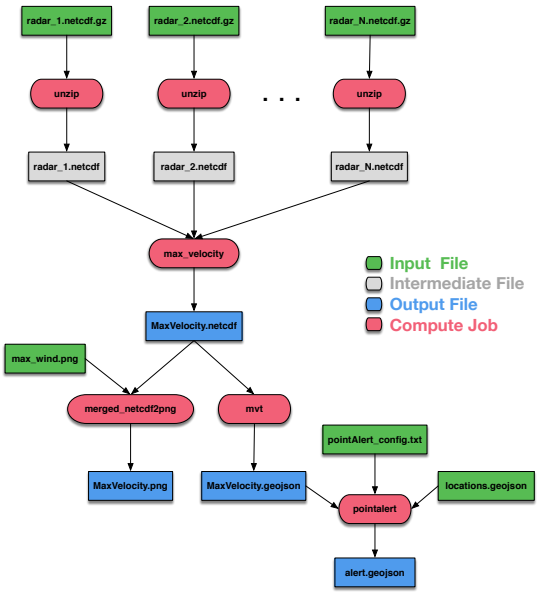

Fig. 5. CASA Wind Pegasus Workflow.

Nowcast. The Pegasus Nowcast workflow [37] computes short-term advection forecasts, as described in Section III-A, by splitting grided reflectivity data into 31 grids and computing reflectivity predictions over the next 30 minutes. An abstract version of the workflow's DAG is presented in Figure 6, which reveals that the size of the workflow doesn't depend on the input, and the number of *compute* tasks is fixed. The nowcast workflow contains 63 compute tasks in total, 1 task for splitting the input data into 31 individual grids, and then 62 independent tasks that compute the reflectivity and the respective contour images. All tasks run within a Singularity container that is managed by Pegasus and has a size of 153MB.

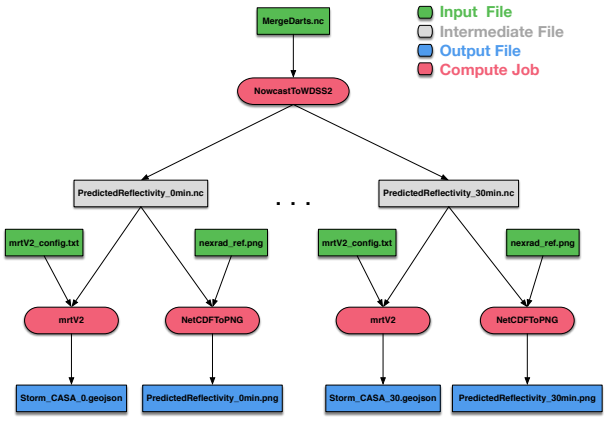

Fig. 6. CASA Nowcast Pegasus Workflow.

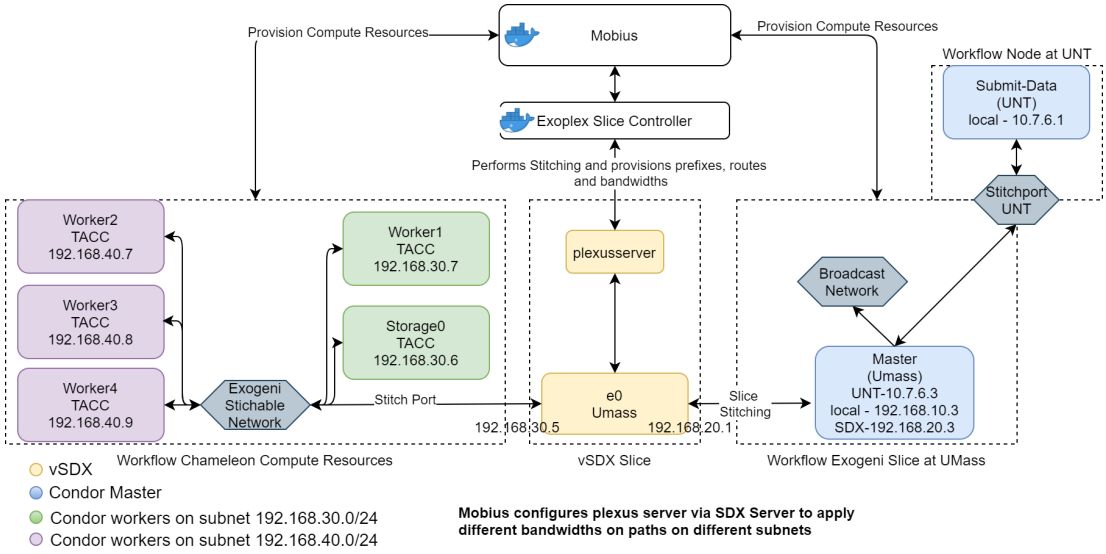

Fig. 7. CASA vSDX workflow deployment.

Wind. The Pegasus Wind Speed workflow [38] computes the maximum wind velocity, by combining multiple single radar output to account for single radar measurement inaccuracies (Section III-B). An abstract version of this workflow's DAG is depicted in Figure 5. To construct the input for the wind speed pipeline (preprocessing phase), single radar data files are accumulated over a variable time window (minimum 1 minute), which regulates how often CASA produces maximum wind velocity contours, but also affects the size of the input of a single workflow run. As a result the first level of tasks (unzipping any zipped files) in the wind speed workflow (Figure 5) depends on the number of input files, and thus the workflow has a variable number of tasks. The unzipping phase is followed by four compute tasks that output the wind products and notify points of interest for severe weather. These four tasks are running within a Singularity container, 163MB in size.

Workflow Testcases. To conduct our evaluation, both workflows are processing 30 minutes of pre-captured real weather data, which we replay as if they were arriving in real-time to simulate a production scenario from CASA's operations. The individual files consumed by the nowcast workflow are 9.6MB in size and the total size is 287MB. On the other hand the dataset for the wind workflows is comprised by files with individual size of ~12MB, and the total dataset size is ~6GB. For the two workflows we replay the data using an accumulation interval of 1 minute and we are using Pegasus-EM to identify the newly added files and queue nowcast or wind workflow to their respective ensembles.

B. Experimental Infrastructure

For evaluation, we used the DyNamo system to deploy a production scenario that is similar to CASA's day to day operational radar data processing setup, and spreads across both ExoGENI and Chameleon testbeds (Figure 7). In our setup Mobius and the vSDX controller are running within Docker containers at our USC Information Sciences Institute (ISI) Docker cluster.

Additionally, we are using one of CASA's operational nodes at the University of North Texas (UNT) in Denton, TX, to host the data and submit the Pegasus workflows. The vSDX nodes and the workflow master node are located on ExoGENI at the University of Massachusetts Amherst (UMass) rack, on separate slices, while the compute nodes are located on Chameleon at TACC. To establish the layer2 connectivity between the sites, Mobius "stitched" the UNT server to the workflow master node and instructed the vSDX controller to stitch the same node to the Chameleon nodes via the vSDX slice. The Chameleon compute cluster contains 5 nodes, 4 compute nodes and 1 storage node. 3 of the compute nodes reside in the 192.168.40.0/24 subnet while the other compute node and the storage node reside in subnet 192.168.30.0/24. Each node has 24 physical cores with hyperthreading (48 threads), 192GB RAM, 250GB SSD and is connected to a shared 10Gbps network. During the experiments we did not use the storage node to optimize for network traffic, but it was used as a next hop to route traffic from the subnet (192.168.40.0/24) that did not match the Chameleon stitchport's subnet.

As we have shown in our initial evaluation of the DyNamo system [2] 144 and 48 HTCondor compute slots are enough to execute the nowcast and the wind speed workflow ensembles, respectively, without any compute imposed delays. Using HTCondor tags, the 3 compute nodes residing on the subnet 192.168.40.0/24 have been assigned to nowcast workflow tasks, while the node on the subnet 192.168.30.0/24 has been assigned to the wind speed workflow. Finally, all the stitchable networks were created with a network bandwidth of 1Gbps.

Software. On the submit node (where parts of the Dynamo system reside), the master node and the worker nodes we

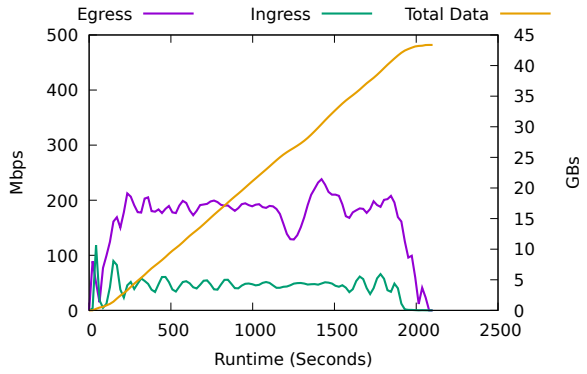

Fig. 8. Wind Ensemble - Network Utilization.

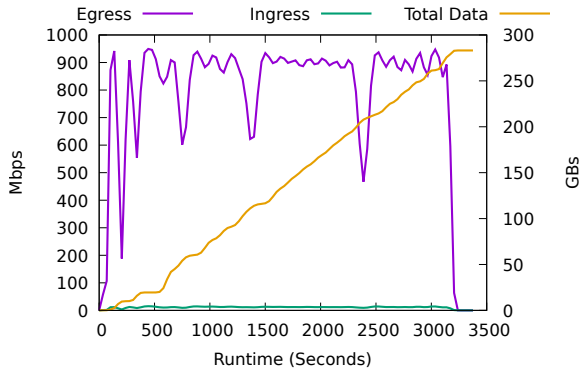

Fig. 9. Nowcast Ensemble - Network Utilization.

have installed HTCondor v8.8.9, and we have customized its configuration to match the role of each node. In this setup, the workers are configured with partitionable slots and they advertise a workflow tag so they can be matched to the correct workflow. Additionally, on the submit node we have installed the nightly build of Pegasus v5.0.0 and the Apache HTTP server, to allow the workers to retrieve input files, configuration files and the application containers over HTTP. All of the workers use Singularity v3.6.1, and Mobius was used to provision compute resources on ExoGENI and Chameleon, and establish the network connections between ExoGENI, Chameleon and the CASA repository.

C. Workflow Ensembles - Network Requirements

The two workflow ensembles present different network requirements due to the amount of tasks and the container transfers they instantiate. We profile the network utilization on CASA's data repository at UNT, during the execution of the two workflow ensembles, using a dedicated 1Gbps layer2 connection and the testcase datasets described in Section V-A.

Figure 8 shows that the wind workflow ensemble is executed for ~2100 seconds, has an average bandwidth usage of ~200Mbps with a peak close to 250Mbps, while the total amount of data transferred is ~44GBs.

Figure 9 depicts the network utilization imposed by the nowcast workflow ensemble. The nowcast calculations are occupying resources for ~3200 seconds and they lead the network to congestion for prolonged periods of time. The average network utilization is close to 900Mbps with spikes reaching 960Mbps, and the total amount of data transferred is ~280GBs.

From Figures 8 and 9 it is clear that the two workflow ensembles cannot fairly share the shame network resources without one of them impacting the other's QoS constraints, since the nowcast workflow ensemble will lead to prolonged network congestion. In our previous work [2] we used workflow runtime optimizations provided by Pegasus (e.g., task clustering) in order to lower nowcast's network requirements, but we did not apply them to this study since it is our goal to evaluate the effectiveness of DyNamo's new network QoS capabilities.

D. Experimental Results

To produce our results we repeated the workflow ensemble executions 5 times, leading to 900 workflow submissions and over 240,000 file transfers generating over 900GBs of network traffic. Figures 10 and 11 present makespan statistics of the individual workflows of the ensembles, while Figures 12 and 13 present statistics of the individual data transfers of the workflow ensembles.

1) Dedicated link performance: To conduct our analysis, we first executed the nowcast and wind workflow ensembles under the best conditions possible, using 1Gbps dedicated layer2 connection. For the wind ensemble Figures 10 and 12 show a very consistent workflow duration (~300 seconds) and file transfer duration with very little deviation. On the other hand, since the nowcast workflow was creating network congestion we observe a noticeable deviation in both the workflow and file transfer durations (Figures 11 and 13). More than half of the workflows in the nowcast ensemble are completed within less than 1500 seconds. However, there are workflow executions that take from 500 seconds all the way to ~2,400 seconds.

2) Uncontrolled network sharing: When we allow the two workflow ensembles to share the same network resources without any QoS policy, then we observe a very noticeable increase to the workflow makespans (Figures 10, 11 middle). The most impacted are the workflows of the wind ensemble, where the average workflow duration increases from 300 seconds to over 1000 seconds, with some workflows completing execution close to 1800 seconds. This is an increase of over 500%. The impact of the additional network overhead is also visible in the nowcast workflows, although more subtle. The median nowcast workflow duration increased by about 200 seconds, while there were more workflows to the far ends of the spectrum.

3) Applying QoS policies: Finally, based on the network profiles presented in Figure 8 and Figure 9 we allocated 300Mbps of the available network bandwidth to the wind workflow ensemble and 700Mbps to the nowcast workflow ensemble, in an attempt to accommodate any network spikes of the wind ensemble. Both Figures 10 and 12 (right) show

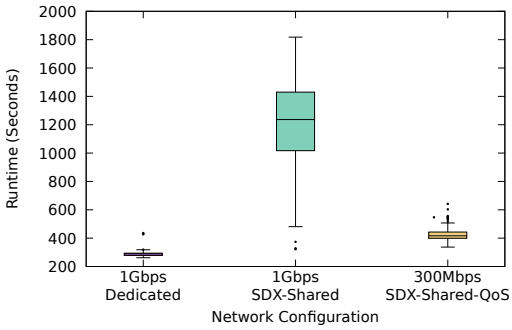

Fig. 10. Wind Ensemble Workflow Makespans.

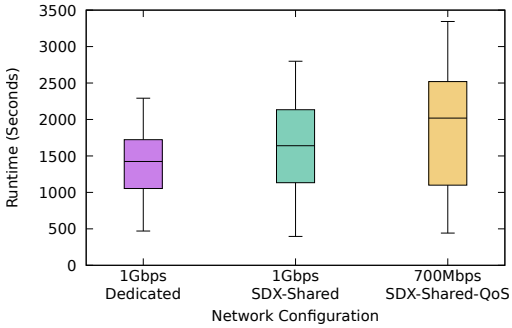

Fig. 11. Nowcast Ensemble - Workflow Makespans.

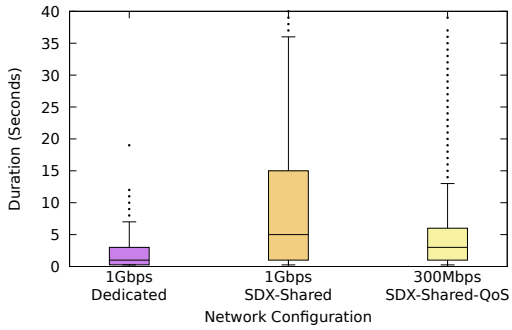

Fig. 12. Wind Ensemble - Workflow Data Transfer Durations.

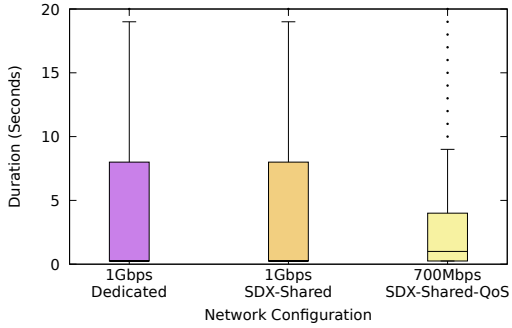

Fig. 13. Nowcast Ensemble - Data Transfer Durations.

an improvement of the wind workflow median makespan and data transfer durations. The wind ensemble's statistics have returned to a more consistent and predictable state with small deviation, similar to the execution conditions when a dedicated network link was used. Meanwhile, as it was expected, the median runtime of the nowcast workflows has increased since there is less available bandwidth (700Mbps) than what the workflow would optimally require (~900Mbps). However, the relative increase in comparison to the dedicated link runtimes is less than 60%. Something we did not expect to see was that even though the median duration of the file transfers in the nowcast ensemble increased by a few seconds, the transfers became more consistent, reducing the duration of the slowest transfers.

From our experiments it is clear that DyNamo can aid to maintain the QoS of workflow ensembles when they are facing uneven network contention. Even though, TCP congestion algorithms attempt to provide a fair share of the network to all of the flows occupying it [39], they cannot provide it at the level of workflow ensembles. Workflow ensembles that flood the network with transfers are claiming a bigger chunk of the available bandwidth, impacting other ensembles with fewer transfers, which struggle to gain their network share. Through well-thought infrastructure deployment-design DyNamo can identify the individual flows that belong to specific workflow ensembles and effectively allocate bandwidth to meet their QoS expectations.

VI. Related Work

There has been extensive prior work on the topics of cloud support for various types of science applications. In this section, we review the related works, which can be classified into three categories: cloud platforms, inter-domain networking and compute infrastructure provisioning for science workflows, and science workflow management systems.

Cloud platforms. A lot of work has been done on the development of research and commercial cloud infrastructures. A number of public cloud providers, such as Amazon EC2 [40] and Microsoft Azure [41], offer IaaS abstractions and some ability to orchestrate them together with networks through mechanisms like CloudFormation [42] and Heat [43]. However, data movement among different cloud providers and infrastructures is expensive and hard to implement, which significantly limits the use of commercial clouds in science applications [44]. The GlobusOnline [45] project provides users the ability to efficiently move data from one computing resource to another, however, it does not provide unified environments for science workloads. In the work presented in this paper, we focus on integration of scalable, reconfigurable distributed testbeds, including ExoGENI [1] and Chameleon [4] with emphasis on data movement.

Inter-domain networking and compute infrastructure provisioning for science workflows. There have been extensive survey papers [46]–[48] in regards to provisioning IaaS cloud resource for scientific workflows. Wang et al. [49] propose an approach to build and run scientific workflows on a

federation of clouds using Kepler and CometCloud. Moreover, there have been strategies for workflow systems to deploy virtual machines in the cloud with limited support for on-demand provisioning and elasticity, while none or minimal support to infrastructure optimization is enabled. Ostermann et al. [50] discussed a set of VM provisioning policies to acquire and release cloud resources for overflow grid jobs from workflows, and characterized the impact of those policies on execution time and overall cost. In prior work [2], we presented dynamic provisioning techniques that spawn resources based on compute elasticity using Mobius [51].

On the perspective of networking between the compute, storage and instrument sites, Macker et al. [52] describe workflow paradigms to address network edge workflow scenarios. Ramakrishnan et al. [53] present experience for virtualized reservations for batch queue systems, as well as coordinated usage of TeraGrid, Amazon EC2 and Eucalyptus (cloud) resources with fault tolerance through automated task replication. Liu et al. [54] developed the Virtual Science Network Environment (VSNE) that emulates the multi-site host and network infrastructure, wherein software can be tested based on mininet with SDN capabilities.

As an important factor, many of the prior works have thrived to achieve a satisfactory Quality of Service (QoS) for the provisioned resources, as indicated by many recent survey papers [55], [56]. Varshney et al. [56] proposed QoS based workload scheduling mechanism by considering energy consumption, execution cost and execution time as QoS parameters. The Department of Energy's ESNet has proposed an On-Demand Secure Circuits and Advance Reservation System [57], which provides software system for booking time and resources on high-speed science networks used by large teams of researchers to share vast amounts of data.

Our work presented in this paper differs from the above by presenting easy-to-use, on-demand resource provisioning mechanisms for malleable data movement and compute provisioning for inter-cloud workflows. We provide dedicated network connections among multiple cloud provider sites with guaranteed performance and QoS.

Science workflow management systems. Several workflow management systems focus on the optimization of science application management on cloud platforms. Islam et al. [58] presented a scalable workflow management system specifically for Hadoop applications. Senturk et al. [59] deal with bioinformatics applications on multi-clouds with a focus on resource provisioning. Malawski et al. [60] presented cost optimization modeling for scheduling workflows on public clouds to minimize the cost of workflow execution under deadline constraints. Abrishami et al. [61] presented workflow scheduling algorithms based on partial critical paths, which also optimize for cost of workflow execution while meeting deadlines. With the rise of multi-clouds, many workflow management systems have focused on this type of platform. Matthew et al. [62] discuss workflow management on multi-cloud brokering among multi-cloud domains with heterogeneous security postures. In this paper, we propose a new approach to enable dynamic resource provisioning, which is integrated with a workflow management system, and demonstrated through deployments with science applications.

VII. Conclusion

We presented three extensions to the DyNamo system that address processing, infrastructure and monitoring challenges of CASA's distributed, atmospheric science workflows. By enhancing Mobius with Virtual Software Define Exchange (vSDX) capabilities, the DyNamo framework can now provide link adaptation, flow prioritization and traffic control between endpoints. Even when single workflow ensembles attempt to create network congestion, other ensembles can maintain their own QoS requirements, avoiding unfair network usage. To evaluate the QoS polices we deployed two of CASA's workflow ensembles (wind speed and nowcast) and we showed that even though the nowcast ensemble is capable of interfering with the wind ensemble, by applying the QoS policies the interference is removed and the wind ensemble performs similarly to as if it was using a dedicated network link. Additionally, we extended the Pegasus Ensemble Manager (Pegasus-EM) to support file and time-based workflow triggering logic that allows CASA to automatically execute its workflows as new data arrive while managing the number of the concurrent workflows being executed. Finally we incorporated the Prometheus monitoring system into the DyNamo framework, providing comprehensive information about the status of the network and the compute resources, allowing CASA scientists to better understand the performance of their provisioned resources across the clouds. In the future, we plan to extend the DyNamo system's capabilities by stitching to more resource providers, supporting streaming workflows, developing new CASA workflows and provide mechanisms that will allow the applications to automatically evaluate the current pressure applied on the provisioned resources and make adjustments to the infrastructure without user intervention (e.g., change the QoS policies of workflow ensembles).

Acknowledgment

This work is funded by NSF award #1826997. We thank Mert Cevik (RENCI), engineers from UNT and LEARN for the UNT stitchport setup. Results in this paper were obtained using Chameleon and ExoGENI testbeds supported by NSF.

References

[1] I. Baldin, J. Chase, Y. Xin, A. Mandal, P. Ruth, C. Castillo, V. Orlikowski, C. Heermann, and J. Mills, *ExoGENI: A Multi-Domain Infrastructure-as-a-Service Testbed*, 2016, pp. 279–315.

[2] E. Lyons, G. Papadimitriou, C. Wang, K. Thareja, P. Ruth, J. Villalobos, I. Rodero, E. Deelman, M. Zink, and A. Mandal, "Toward a dynamic network-centric distributed cloud platform for scientific workflows: A case study for adaptive weather sensing," in *2019 15th International Conference on eScience (eScience)*, 2019, pp. 67–76.

[3] A. Gupta, L. Vanbever, M. Shahbaz, S. P. Donovan, B. Schlinker, N. Feamster, J. Rexford, S. Shenker, R. Clark, and E. Katz-Bassett, "Sdx: A software defined internet exchange," in *SIGCOMM*, 2014, p. 551–562.

[4] J. Mambretti, J. Chen, and F. Yeh, "Next generation clouds, the chameleon cloud testbed, and software defined networking (sdn)," in *2015 International Conference on Cloud Computing Research and Innovation (ICCCRI)*, Oct 2015, pp. 73–79.

[5] E. Deelman, K. Vahi, G. Juve, M. Rynge, S. Callaghan, P. J. Maechling, R. Mayani, W. Chen, R. Ferreira da Silva, M. Livny, and K. Wenger, "Pegasus: a workflow management system for science automation," *Future Generation Computer Systems*, vol. 46, pp. 17–35, 2015, funding Acknowledgements: NSF ACI SDCI 0722019, NSF ACI SI2-SSI 1148515 and NSF OCI-1053575. [Online]. Available: http://pegasus.isi.edu/publications/2014/2014-fgcs-deelman.pdf

[6] "National Energy Research Scientific Computing Center (NERSC)," https://www.nersc.gov.

[7] "Oak Ridge Leadership Computing Facility," https://www.olcf.ornl.gov.

[8] J. Towns, T. Cockerill, M. Dahan, I. Foster, K. Gaither, A. Grimshaw, V. Hazlewood, S. Lathrop, D. Lifka, G. D. Peterson, R. Roskies, J. Scott, and N. Wilkins-Diehr, "Xsede: Accelerating scientific discovery," *Computing in Science & Engineering*, vol. 16, no. 05, pp. 62–74, sep 2014.

[9] R. Pordes, D. Petravick, B. Kramer, D. Olson, M. Livny, A. Roy, P. Avery, K. Blackburn, T. Wenaus, F. Würthwein, I. Foster, R. Gardner, M. Wilde, A. Blatecky, J. McGee, and R. Quick, "The open science grid," *Journal of Physics: Conference Series*, vol. 78, p. 012057, jul 2007. [Online]. Available: https://doi.org/10.1088\%2F1742-6596\%2F78\%2F1\%2F012057

[10] K. Keahey, P. Riteau, D. Stanzione, T. Cockerill, J. Mambretti, P. Rad, and P. Ruth, "Chameleon: a scalable production testbed for computer science research," in *Contemporary High Performance Computing: From Petascale toward Exascale*, 1st ed., ser. Chapman & Hall/CRC Computational Science, J. Vetter, Ed. Boca Raton, FL: CRC Press, 2018, vol. 3, ch. 5.

[11] D. Thain, T. Tannenbaum, and M. Livny, "Distributed computing in practice: the condor experience," *Concurr. Comput.*, vol. 17, no. 2-4, pp. 323–356, Feb. 2005.

[12] D. Gunter, E. Deelman, T. Samak, C. Brooks, M. Goode, G. Juve, G. Mehta, P. Moraes, F. Silva, M. Swany, and K. Vahi, "Online workflow management and performance analysis with stampede," in *7th International Conference on Network and Service Management (CNSM-2011)*, 2011.

[13] A. Bayucan, R. L. Henderson, C. Lesiak, B. Mann, T. Proett, and D. Tweten, "Portable batch system: External reference specification," in *Technical report, MRJ Technology Solutions*, vol. 5, 1999.

[14] "Simple Linux Utility for Resource Management." [Online]. Available: http://slurm.schedmd.com/

[15] R. Raman, M. Livny, and M. Solomon, "Matchmaking: distributed resource management for high throughput computing," in *Proceedings. The Seventh International Symposium on High Performance Distributed Computing (Cat. No.98TB100244)*, 1998, pp. 140–146.

[16] J. Frey, T. Tannenbaum, I. Foster, M. Livny, and S. Tuecke, "Condor-G: A computation management agent for multi-institutional grids," in *Proceedings of the Tenth IEEE Symposium on High Performance Distributed Computing (HPDC)*, San Francisco, California, August 2001, pp. 7–9.

[17] Mobius Github Repository, https://github.com/RENCI-NRIG/Mobius.

[18] "Internet 2," https://www.internet2.edu/.

[19] "The energy science network," https://www.es.net/.

[20] "Openstack." [Online]. Available: https://www.openstack.org/

[21] D. McLaughlin, D. Pepyne, V. Chandrasekar, B. Philips, J. Kurose, M. Zink, K. Droegemeier, S. Cruz-Pol, F. Junyent, J. Brotzge, D. Westbrook, N. Bharadwaj, Y. Wang, E. Lyons, K. Hondl, Y. Liu, E. Knapp, M. Xue, A. Hopf, K. Kloesel, A. DeFonzo, P. Kollias, K. Brewster, R. Contreras, B. Dolan, T. Djaferis, E. Insanic, S. Frasier, and F. Carr, "Short-wavelength technology and the potential for distributed networks of small radar systems," *Bulletin of the American Meteorological Society*, vol. 90, no. 12, pp. 1797–1818, 2009. [Online]. Available: https://doi.org/10.1175/2009BAMS2507.1

[22] E. J. Lyons, M. Zink, and B. Philips, "Efficient data processing with exogeni for the casa dfw urban testbed," in *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, July 2017, pp. 5977–5980.

[23] L. Li, W. Schmid, and J. Joss, "Nowcasting of motion and growth of precipitation with radar over a complex orography," *Journal of Applied Meteorology*, vol. 34, no. 6, pp. 1286–1300, 1995. [Online]. Available: https://doi.org/10.1175/1520-0450(1995)034⟨1286:NOMAGO⟩2.0.CO;2

[24] E. Ruzanski and V. Chandrasekar, "Weather radar data interpolation using a kernel-based lagrangian nowcasting technique," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 6, pp. 3073–3083, June 2015.

[25] Y. Yao, Q. Cao, R. Farias, J. Chase, V. Orlikowski, P. Ruth, M. Cevik, C. Wang, and N. Buraglio, "Toward live inter-domain network services on the exogeni testbed," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2018, pp. 772–777.

[26] Zeek Github Repository, https://github.com/zeek/zeek.

[27] Ahab Github Repository, https://github.com/RENCI-NRIG/ahab.

[28] Linux Foundation Collaborative Projects, https://www.openvswitch.org/.

[29] Open flow SDN Controllers, https://en.wikipedia.org/wiki/List_of_SDN_controller_software/.

[30] Ryu SDN Controller, https://ryu-sdn.org/.

[31] Ryu Rest Router, https://github.com/faucetsdn/ryu/blob/master/ryu/app/rest_router.py.

[32] Exoplex Github Repository, https://github.com/RENCI-NRIG/CICI-SAFE.

[33] S. Pandey, K. Vahi, R. Ferreira da Silva, E. Deelman, M. Jian, C. Harrison, A. Chu, and H. Casanova, "Event-based triggering and management of scientific workflow ensembles," 2018, poster presented at the HPC Asia 2018: Tokyo, Japan. [Online]. Available: http://sighpc.ipsj.or.jp/HPCAsia2018/poster/post102s2-file1.pdf

[34] Prometheus, https://prometheus.io/.

[35] Node Exporter, https://prometheus.io/docs/guides/node-exporter/.

[36] Grafana, https://grafana.com/.

[37] Scitech, "CASA Nowcast Pegasus Workflow," https://github.com/pegasus-isi/casa-nowcast-workflow.

[38] ——, "CASA Wind Pegasus Workflow," https://github.com/pegasus-isi/casa-wind-workflow.

[39] G. Hasegawa, M. Murata, and H. Miyahara, "Fairness and stability of congestion control mechanisms of tcp," in *IEEE INFOCOM '99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No.99CH36320)*, vol. 3, 1999, pp. 1329–1336 vol.3.

[40] Amazon Elastic Compute Cloud, http://www.amazon.com/ec2.

[41] Microsoft Azure Cloud, https://azure.microsoft.com/en-us/.

[42] AWS CloudFormation, http://aws.amazon.com/cloudformation.

[43] OpenStack Heat Project, https://wiki.openstack.org/wiki/Heat.

[44] I. Baldin, P. Ruth, C. Wang, and J. S. Chase, "The future of multi-clouds: A survey of essential architectural elements," in *2018 International Scientific and Technical Conference Modern Computer Network Technologies (MoNeTeC)*, Oct 2018, pp. 1–13.

[45] I. Foster, "Globus online: Accelerating and democratizing science through cloud-based services," *IEEE Internet Computing*, vol. 15, no. 3, pp. 70–73, May 2011.

[46] J. Liu, E. Pacitti, P. Valduriez, and M. Mattoso, "A survey of data-intensive scientific workflow management," *Journal of Grid Computing*, vol. 13, no. 4, pp. 457–493, Dec. 2015.

[47] G. Galante, L. C. Erpen De Bona, A. R. Mury, B. Schulze, and R. Rosa Righi, "An analysis of public clouds elasticity in the execution of scientific applications: A survey," *Journal of Grid Computing*, vol. 14, no. 2, pp. 193–216, Jun. 2016.

[48] E. F. Coutinho, F. R. de Carvalho Sousa, P. A. L. Rego, D. G. Gomes, and J. N. de Souza, "Elasticity in cloud computing: a survey," *annals of telecommunications - annales des télécommunications*, vol. 70, no. 7, pp. 289–309, Aug 2015.

[49] J. Wang, M. AbdelBaky, J. Diaz-Montes, S. Purawat, M. Parashar, and I. Altintas, "Kepler + cometcloud: Dynamic scientific workflow execution on federated cloud resources," *Procedia Computer Science*, vol. 80, pp. 700 – 711, 2016, international Conference on Computational Science 2016, ICCS 2016, 6-8 June 2016, San Diego, California, USA.

[50] S. Ostermann, R. Prodan, and T. Fahringer, "Dynamic cloud provisioning for scientific grid workflows," in *2010 11th IEEE/ACM International Conference on Grid Computing*, Oct 2010, pp. 97–104.

[51] A. Mandal, P. Ruth, I. Baldin, Y. Xin, C. Castillo, G. Juve, M. Rynge, E. Deelman, and J. Chase, "Adapting scientific workflows on networked clouds using proactive introspection," in *IEEE/ACM Utility and Cloud Computing (UCC)*, 2015.

[52] J. P. Macker and I. Taylor, "Orchestration and analysis of decentralized workflows within heterogeneous networking infrastructures," *Future Generation Computer Systems*, vol. 75, pp. 388 – 401, 2017.

[53] L. Ramakrishnan, C. Koelbel, Y. Kee, R. Wolski, D. Nurmi, D. Gannon, G. Obertelli, A. YarKhan, A. Mandal, T. M. Huang, K. Thyagaraja, and D. Zagorodnov, "Vgrads: enabling e-science workflows on grids and clouds with fault tolerance," in *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, 2009, pp. 1–12.

[54] Q. Liu, N. S. V. Rao, S. Sen, B. W. Settlemyer, H.-B. Chen, J. M. Boley, R. Kettimuthu, and D. Katramatos, "Virtual environment for testing software-defined networking solutions for scientific workflows," ser. AI-Science'18, 2018.

[55] M. H. Ghahramani, M. Zhou, and C. T. Hon, "Toward cloud computing qos architecture: analysis of cloud systems and cloud services," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 1, pp. 6–18, 2017.

[56] S. Varshney, R. Sandhu, and P. K. Gupta, "Qos based resource provisioning in cloud computing environment: A technical survey," in *Advances in Computing and Data Sciences*, M. Singh, P. Gupta, V. Tyagi, J. Flusser, T. Ören, and R. Kashyap, Eds., 2019, pp. 711–723.

[57] "On-demand secure circuits and advance reservation system," https://www.es.net/engineering-services/oscars/.

[58] M. Islam, A. K. Huang, M. Battisha, M. Chiang, S. Srinivasan, C. Peters, A. Neumann, and A. Abdelnur, "Oozie: Towards a scalable workflow management system for hadoop," ser. SWEET '12, 2012.

[59] I. F. Senturk, P. Balakrishnan, A. Abu-Doleh, K. Kaya, Q. Malluhi, and Ümit V. Çatalyürek, "A resource provisioning framework for bioinformatics applications in multi-cloud environments," *Future Generation Computer Systems*, vol. 78, pp. 379 – 391, 2018.

[60] M. Malawski, K. Figiela, M. Bubak, E. Deelman, and J. Nabrzyski, "Scheduling multilevel deadline-constrained scientific workflows on clouds based on cost optimization," *Scientific Programming*, vol. 29, pp. 158–169, Jan. 2015.

[61] S. Abrishami, M. Naghibzadeh, and D. H. Epema, "Deadline-constrained workflow scheduling algorithms for infrastructure as a service clouds," *Future Generation Computer Systems*, vol. 29, no. 1, pp. 158 – 169, 2013.

[62] M. Dickinson, S. Debroy, P. Calyam, S. Valluripally, Y. Zhang, R. Bazan Antequera, T. Joshi, T. White, and D. Xu, "Multi-cloud performance and security driven federated workflow management," *IEEE Transactions on Cloud Computing*, pp. 1–1, 2018.