Data Sci. Eng. (2016) 1(2):65–72
DOI 10.1007/s41019-016-0008-y

INVITED PAPER

Software-Defined Storage-Based Data Infrastructure Supportive of Hydroclimatology Simulation Containers: A Survey

Wonjun Lee[1] · **Sanjiv Kumar**[2]

Received: 6 April 2016 / Revised: 8 June 2016 / Accepted: 10 June 2016 / Published online: 1 July 2016

Abstract Hydroclimatic research requires highly intensive resources in terms of computation and data to perform simulations. Setting up complex experiment environment and configurations to submit jobs in computational clusters as well as managing user's limited storage spaces by transferring big size data into the secondary storage are complicated and time-consuming. As a possible answer to address such issues in hydroclimatic research, new technologies, software-defined storage and containers have been introduced. When the two technologies are combined to support hydroclimatic simulations, we discuss how the software-defined storage data infrastructure strengthens containers in terms of flexibility of data handling and storage scalability.

Keywords Hydroclimatology · Software-defined storage · Container · Scalability

1 Introduction

The hydroclimatology simulation models have evolved over the past 10 years becoming larger and more complex. The community earth system model (CESM) [1] is a widely used global climate model that allows researchers to conduct fundamental research about the Earth's past, present, and future climate states. The CESM is composed of five separate models simultaneously simulating the Earth's atmosphere, ocean, land surface, land-ice, and sea-ice, and one central coupler component. CESM simulations require intensive resources, i.e., big data storage as well as high-performance computing for large and complex simulations. The global land–atmosphere coupling experiments (GLACE) [2] Hydrology experiment (described later) uses CESM climate simulations to investigate role of land processes, e.g., soil moisture and precipitation interactions on drought and flood conditions. Experiment has been performed at 1-degree spatial resolution generating 1 TB data per ensemble member totaling 20 TB for 20 ensembles. Ensemble climate simulations are necessary to separate signal from the noise due to internal climate variability [3].

There are several projects facilitating climate data and increasing model interoperability: earth system grid (ESG), earth system modeling framework (ESMF), earth system curator, EU-METAFOR, and Purdue CCSM [4–8]. The ESG provides data and knowledge management system built on the data infrastructure that covers over 500 TB of data collections for climate model comparisons and supports a total download volume of over a petabyte to over 20,000 registered users [4]. The TeraGrid science gateway program developed a community climate system modeling (CCSM) portal that aims at bringing large parallel climate model simulations to students in the classrooms and to educators and beginners in climatic research who are not familiar with high-performance computing (HPC) [9]. As a result, the CCSM portal support 25 students in a class submitting 1310 jobs that use total 162,000 processor hours and generate total 0.6 TB output data [9].

As hydroclimatic simulations involve huge sized data, the high volume of storage is required. In order to run many simulations in a limited working space or to save the output data for future analysis, users generally move output data, which are stored in a scratch disk space, to

✉ Wonjun Lee
wonjun.lee@utsa.edu

Sanjiv Kumar
sanjiv.kumar@noaa.gov

1 University of Texas at San Antonio, San Antonio, TX, USA

2 NOAA/ESRL, Physical Science Division, Boulder, CO, USA

storage such as tape storages or data grid for example, iRODS [10].

There has been a demand for use of unlimited storage by researchers, who want to investigate fundamental climate processes without time for dealing with data and storage, and by classroom students to get hands-on experience in running complex climate model simulations. With such demands, one of the most critical issues the storage administrators face is to construct and manage data infrastructure by eliminating or hiding the complexity from the end-users while maintaining flexibility, scalability, and security. To address such issues, a new paradigm called software-defined storage (SDS) recently proposed. The main goal of the SDS is to provide data storage space to end-users dynamically and flexibly while hiding the complexity of the storage resource management.

In addition to SDS, many new technologies for HPC have emerged. Recently, many IT departments in industry and organizations have been trying to convert their computing platforms from hardware-based virtual machines to the traditional Linux or CoreOS [11]-based Docker [12] containers. A recent survey reports that 70 % of the respondents are using or evaluating Docker containers mostly in QA/test and development [13]. The container technology helps even hydroclimatic research community because of the container's strengths, i.e., high scalability, easy deployment, and easy portability. CESM model runs require highly scalable compute nodes to be run in parallel to reduce the execution time as well as data processing time for input and output data. In addition, researchers who want to deploy the CESM model for their research should go through all the steps of installation of the required software with appropriate configurations on different platform. Such efforts of setting up scalable experiment environment and deploying the CESM model are challenging and time-consuming. Thus, the hydroclimatic research tools would also require architectural shift to container technology in near future. The SDS supports such container-based hydroclimatic simulations by ensuring fast, and flexible access to persistent data storage and high scalability of storage services when simulation runs scale from a few nodes to thousands. Additionally, the SDS should support containers by providing different quality levels of storage services according to the different purpose of container images.

The rest of the paper is organized as follows: A GLACE-hydrology experiment as a sample hydroclimatic simulation is introduced in Sect. 2 followed by the description of the SDS data infrastructure in terms of hydroclimatic research in Sect. 3. Section 4 describes container technology for hydroclimatic research. Section 5 describes the integration of SDS and container technologies. Section 6 concludes the paper.

2 Hydroclimatic Simulation (The GLACE-Hydrology Experiment)

2.1 Experiment Overview

Ongoing California Drought, 2013 Boulder Flood, and 2015 Texas Flood are just few of numerous examples that emphasize improving our understanding and predictability of hydroclimatic extremes. While these events are large-scale anomalous weather patterns, e.g., Northwest Pacific Ridge identified as the potential cause of ongoing drought in California, land-atmosphere interactions can accentuate or ameliorate intensity and duration of hydroclimatic extremes though feedbacks to regional atmospheric processes such as recycling of locally evaporated moisture, convection triggering mechanism, and nonlinear hydrological response processes, e.g., infiltration rate, and runoff generation [14–22]. Existing/standard climate simulations do not allow quantify the role of land–atmosphere interactions for which two parallel runs are required: (a) coupled land–atmosphere run with time-evolving specified sea surface temperature, also known as standard atmospheric model inter-comparison project (AMIP) simulation, and (b) uncoupled land–atmosphere runs standard AMIP run with specified soil moisture climatology and time-evolving sea surface temperature (land uncoupled AMIP simulation), thus effectively decoupling synchronous land surface feedback to the atmosphere. The difference between these two experiments is the effects of coupling on climate predictability [23]. The experiment utilizes GLACE framework for hydroclimatology research [2]. The overall goal of this study is to improve representation of hydroclimatic extremes in the current generation of earth system models by comparing model results with observations or equivalent data and emphasizing the roles of land hydrological processes and its interactions with the atmospheric processes.

2.2 Numerical Approach

This study uses community earth system model developed at NCAR to study role of land-atmosphere interactions on hydroclimatic extremes [24]. In addition to standard AMIP and land uncoupled AMIP simulations described earlier, offline land model simulations are needed to investigate the impacts of land–atmosphere coupling on hydrological processes. The offline land model simulations are performed using 3-hourly atmospheric data generated by AMIP simulations.

2.3 Computational Experiment

Experiment 0 (E0): Observations, no core hours needed.

Table 1 Experiments using observed SST

Experiment	Length in years (1971 to 2015)	Number of ensemble	Peak hours per year and per ensemble	Row total
E1-W	45	10	1070	481,500
E1-S	45	10	1070	481,500
E2-W	45	10	185	83,250
E2-S	45	10	185	83,250
		Total core hours		1,129,500

Experiment 1 (E1): Intermediate simulations to decouple atmosphere from land. This consists of two sets of parallel AMIP type runs: standard AMIP runs (E1-W), and land uncoupled AMIP simulations (E1-S). Three-hourly data for surface variables (temperature, pressure, wind, humidity, and precipitation) is saved to run offline Land Model simulations. The difference between E1-W and E1-S is the effects of coupling on climate predictability.
Experiment 2 (E2): Two sets of parallel offline land model simulations—one set (E2-W) uses atmospheric forcing data from E1-W and another set (E2-S) uses atmospheric forcing data from E1-S, in which land surface synchronous feedback has been erased or 'un-land-ified.' The difference between E2-W and E2-S is the effects of coupling on hydrological predictability.

All simulations are performed at 1-degree resolution (0.9 × 1.25_gx1v6) and list of all experiments is tabulated (Table 1).

Twenty-member ensemble are initialized using slightly perturbed atmospheric initial conditions [25].

2.4 Simulation Steps

A GLACE-hydrology experiments consist of the steps shown in Fig. 1. At the first step, users create a new simulation case specifying 4 required arguments—case name, machine name, dataset resolution, and component set. From the specified parameters, scripts within the CESM code create a case directory that contains files necessary for the second step. The next step is to configure the case. The users at this step configure specific parameters such as time-span of simulation, trace generation, and the frequency of data.

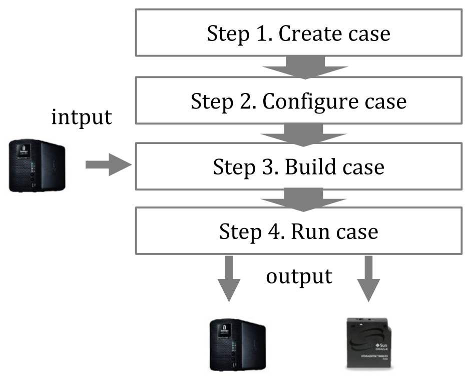

Fig. 1 Simulation steps

In the third step, users invoke scripts to build libraries and executable, and prestage the standard input data required by the simulation. The final step runs the case and creates output files and log files. At the end of the run, output files and log files are archived in a short-term directory.

2.5 Data Management Plan

Each ensemble member of E1-W, and E1-S generates approximately 1 TB data that need to be saved for running the experiments E2-W, and E2-S. These data are also valuable for future analysis. However, such a large volume of data (∼20 TB) can be stored in working directory for 2 months (permission for scratch space). Hence, these data are being moved back and forth to tape storage that takes significant amount of time.

3 Data Infrastructure as a Software-Defined Storage

With the rapid increase in storage demands to process huge hydroclimatology simulation data, many storage administrators see that traditional storage techniques are not suitable to efficiently deal with large data. A new paradigm called SDS recently introduced to address challenges. SDS performs the automation and pooling of storage through a software control unit offering a significant simplification to provisioning and managing. Typically storage is managed as individual products leading to complex and separate management and deployment. However, with SDS, applications have the opportunity to be provisioned the best storage system across different hardware products.

Compared to the traditional storage system, SDS systems have the following differences.

1. Policy-based storage service—users can specify policies about how the underlying storage for the server should

be consumed in terms of availability, reliability, latency, etc. The policies can encompass the number of failures to tolerate, performance, backup period, retention times, and forcing provisioning. For example, if the number of failure to tolerate is set to 1, the storage management system will copy data generated by that server in a physically separated storage [26].

2. Scale-out-based architecture—the storage infrastructure can be supplemented without disrupting availability and performance specified in QoS and service level agreement (SLA). In addition to storage infrastructure, storage services can be provided based upon requests from scaled applications or servers correspondingly.
3. Resource pooling—all storage resources are collected in a logical place and clustered into multiple groups called a pool by the central control unit. Pooling storage resources allows administrators not to deal with individual storage and specific hardware configurations.
4. Abstraction—heterogeneous storage resources are gathered into logical pools where they are consumed and managed in contrast to traditional storage where the control is decentralized.
5. Automatic response—upon user's request, the operations on the SDS system are automatically performed.
6. Transparency—various APIs are provided to storage service users for better visibility into the resources.

SDS is composed of three layers as depicted in Fig. 2.

The application layer at the top represents various types of applications, web or application or database servers that perform transactions on data. In CESM experiments, each job submitted to underlying infrastructure can perform its operations on data by using the provided self-service interfaces via REST APIs and catalogs. For more flexible storage services, the applications or service users can be provided a form of tagging or policies for the type of storage and data services applied [27].

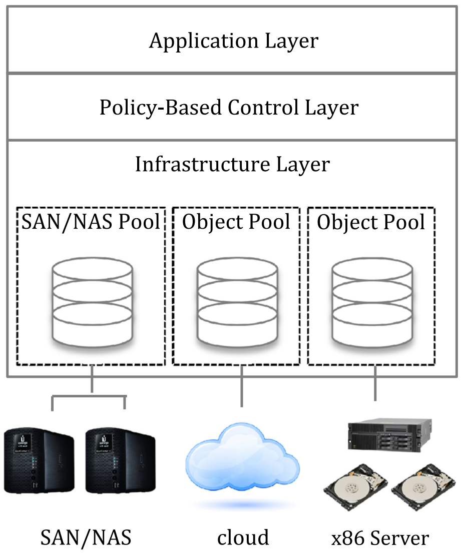

Fig. 2 Architecture of software-defined storage

The control layer is a software component which is core of the SDS. The control software or control unit manages and controls data placed at storage systems in the control layer, which is separated from the infrastructure of the storage assets. This separation reduces management complexity by performing storage configuration, allocation and placement on a centralized control layer. The control layer applies various policies to comply with requests from application layer regardless of vendor, type, and model. In order to apply policies and the service level agreement (SLA), the control unit communicates with application, orchestrator, and storage systems. Applications request storage resources logically located at infrastructure layer through the control unit. To control the storage resources for various requests from applications, control layer uses control operations. A recent SDS experimental framework [28] introduced control functions such as *Get_Number_of_StoredFiles, isFull, Used_space, addFile,* and *Available_space*. By using such functions, the control layer manages storage resources based on the information sent from the storage host. The information is kept in a table referred to Function_Table [28] that includes the up-to-date status of storage devices. While the control layer is monitoring storage resources, if there is a change in availability, Function_Table is updated.

The infrastructure layer combines various storage devices and classifies storage arrays and then binds into pools. Thus, the actual abstraction and pooling of the storage infrastructure are performed in the infrastructure layer.

4 Hydroclimatology Simulations with Container Technology

The hydroclimatology simulation programs have many implicit dependencies on programs, libraries, and other components. As a consequence, a simulation case created and built in one environment does not run correctly in another environment without significant efforts.

In the past, hardware virtualization was used as an answer, but this approach has significant problems. A virtualized machine requires a complete operating system (OS) to run applications in virtualized hardware resulting in very large sized (several gigabytes) image. Whenever a file is added or modified, an overall image including OS should be deployed again.

As an answer to this issue, Docker container technology [12] has recently been proposed. A container is a self-contained package where all components required running the application are included except OS. The package can be efficiently distributed and run across a wide range of computing platforms.

Many scientific and engineering communities have started to use Linux Containers in the cloud to simplify simulations. One of the most popular implementations is UberCloud containers [29]. UberCloud containers are ready-to-execute software packages designed to deliver tools that an engineer needs for completing his task. Engineers can simulate engineering experiments, for example visualization of wind tunnel flows around bicycle riders, from these containers without having to worry about the software details. UberCloud containers are launched from prebuilt images, which are distributed through a central registry hosted by UberCloud. UberCloud container is composed of following software required to complete a particular task.

- Libraries commonly needed by engineers such as MPI which has been installed and configured
- Utilities needed by engineers such as screen sharing applications, diff utilities, and remote visualization applications
- Cloud data storage connectors to multiple cloud storage service providers such as Amazon S3
- Engineering applications, which are preinstalled, tuned and tested within UberCloud containers. An example of engineering applications is the fluid dynamics OpenFOAM software.
- Administration tools such as automated password generation, log monitoring

As UberCloud container image wraps everything that an engineer needs to complete a task, a container image for the CESM tool includes everything that is required to run the simulations. The CESM user's document [4] states that the external system and software for installing and running CESM is as follows.

- csh, sh, and perl scripting languages
- subversion client version 1.4.2 or greater
- Fortran (2003 recommended, 90 required) and C compilers. pgi, intel, and xlf are recommended compilers
- MPI (although CESM does not absolutely require it for running on one processor)
- NetCDF 4.2.0 or newer
- ESMF 5.2.0 or newer (optional)
- pnetcdf 1.2.0 required and 1.3.1 recommended
- Trilinos required for certain configurations
- LAPACKm or a vendor supplied equivalent required for some configurations
- CMake 2.8.6 or newer10 required for configurations that include CISM.

In prebuilt container images, software dependency issues have already been resolved and adjusted. By using prebuilt image that contains installation of various software required to run the CESM models and complex dependencies, hydroclimatic researchers can run their same simulation cases on different platforms. The prebuilt images are distributed through a central registry hosted by each organization based on the information about case name, machine name, dataset resolution, and component set. Updates including bug fixes and software enhancement become fairly fast and available through the central repository.

Besides UberCloud containers for engineers, container technology is being used to help molecular scientists [30]. In [30], scientists utilize *autodock3*, which is a molecular modeling simulation software used for protein–ligand docking, both in hardware-based virtual machines and in Docker containers. Results from the experiments show that the container-based system is more efficient in reducing the overall execution time for molecular applications and has better memory management for multiple containers than hardware-based virtual machines [30].

When using multiple containers for parallel execution of tool, it is important to make sure that they are efficiently used with spikes. Currently the most frequently used cluster managing tools are Kubernetes [31] and Swarm [32]. Kubernetes developed by Google is a container orchestration tool organizing and networking containers. The primary concept of Kubernetes is pod, which is a group of containers that are deployed and scheduled together. A pod typically includes 1 to 5 containers, which work together to provide a service. Containers within a pod share an IP address and thus communicate by using ports on the local–host address.

Swarm is the native clustering tool for Docker. In Swarm, each host runs a swarm agent and one host runs a swarm manager. The manager is in charge of orchestrating and scheduling containers on the hosts. Swarm can be run in a high-availability mode to handle failover. The default method to find and add hosts to a cluster is by using a token that is a list of addresses of hosts stored on the Docker-Hub [33].

5 Integration of SDS and Container

When the data storage infrastructure of CESM simulation platform is constructed based on the SDS and the CESM simulations run based on the containers, then the data storage infrastructure should support containers in a way that strengthens the container's useful features.

SDS-based data storage infrastructure supports container-based CESM applications as follows.

1. It increments the flexibility for containers' data handling
2. It provides highly scalable and high-performance persistent storages accordingly as containers scale out
3. It supports simple container mobility

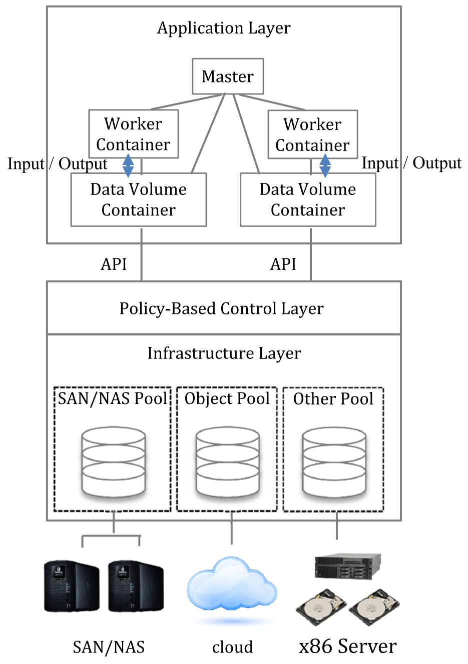

Fig. 3 Software-defined storage architecture supportive of clustered containers

4. It expands the choice of storage platforms for administrators looking to deploy containers which require data persistence

5.1 Providing Flexible Storages for Containers' Data Handling

In application layer, tasks of CESM simulation are split up to multiple containers and run simultaneously with different input in order to obtain results faster. The containers that run a part of the simulation in parallel are called as *worker containers* (Fig. 3). Each worker container concurrently reads input data from and writes output data into a data-only container, which is called as a *data volume container* [13]. The data volumes required by worker container logically exist in data volume container and are mounted to worker container. The data volume containers make possible worker container preserve the generated data and share data volumes even though the data do not physically exist in the host.

When a data volume container is created from a prebuilt image, it initially calls the API to connect to the SDS control layer and to request storage resources. These call routines including some security functions may have already been built into the image. After performing some authentication and authorization processes, the data volume container is assigned a logical volume from the SDS control unit. The service connectivity and resource request between application layer and control layer can be made through the management software and service catalogs or policies specified by

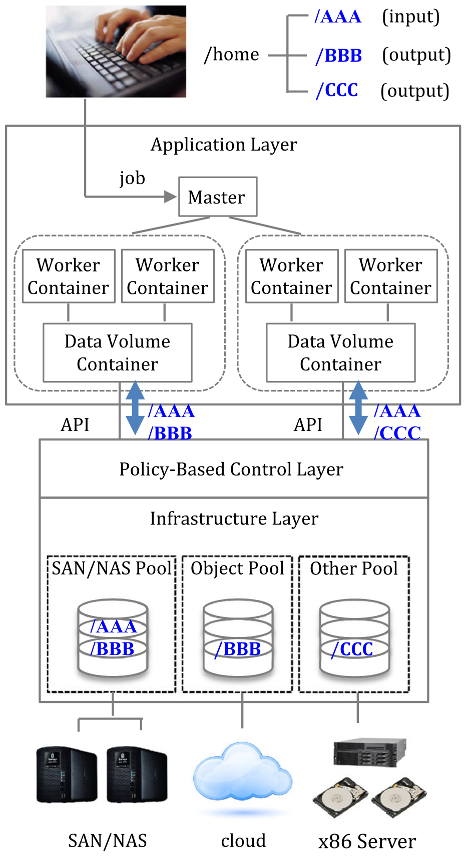

Fig. 4 Logical storage distribution

the master or data volume containers. In order to properly respond to a request from the data volume container for storage resources, SDS storage control unit intelligently selects and places resources based on the specifications.

As the CESM simulations run in a cluster, the container orchestration should also be integrated with SDS. When using the architecture of Kubernetes as an orchestration tool, the worker containers and the data volume container can be deployed and scheduled together at the same pod to better communicate to each other with the port and to share the volume for a specific directory as shown in Fig. 4. In CESM experiment, the specific storage for output data volumes related to land and atmosphere history introduced in Sect. 2, can be shared by worker containers in a pod and can be seen a user as the /BBB and /CCC in Fig. 4. Even though the directories represented as /BBB and /CCC can be seen as a unified volume under a user's directory, the logical locations are spread over the different types of the pools based on the policies. If the output data are used for the next case run or post-processing, they may be stored at a scratch space, say, /CCC. For the output data to be used later, /BBB can be at the object pool consisting of cloud based object storages as well as at a SAN/NAS pool even though users are not aware of underlying infrastructures. The control unit should also deal with the input data by providing an API to data volume container, so that every worker containers that belong to a master node can access the same input data volumes located at a specific place such as SAN/NAS pool. This location is only seen to user as the /AAA in home or working directory.

5.2 Supporting High Scalability of Containers

SDS supports data volume containers by elastically scaling out/in storage resources corresponding to the scaling out/in of data volume containers. The time taken to launch 30,000 containers in 1000 nodes by using Docker Swarm is only less than half a second for each container according to an experiment of Docker engineering team [34]. To support such high scalability of container, the storage resources should be provisioned without any disruption to availability or performance, and any overhead for deciding the storage location to place data based on policies. Since the unique job of control layer in the SDS architecture is providing storage services to applications at any scale, SDS perfectly fits the hydroclimatic applications.

5.3 Supporting High Mobility of Containers

By integrating with SDS, containers can freely move from host to host irrespective of the storage make, model, and vendor. When a data volume container launched on a host that uses traditional storage systems, moves to another host, the persistent data generated by the data volume container can be lost because there is no centralized control unit to manage the mappings between containers and storages. However, in SDS supportive of container systems, because storage devices are abstracted and used by containers across hosts in the pools being controlled by a software control unit, physically moving containers across different hosts does not affect logical locations of the previously saved persistent data.

5.4 Providing Storage Options to Containers

The user of the application containers can make a choice for storage platform based on the type of applications. According to the importance or priority of the container images that contain CESM simulations, storage platforms composed of different levels of pools in terms of performance, data protection, and availability for output data, can be selected. One SDS model proposed by Storage Networking Industry Association [27] shows three types of services for storage platforms, i.e., Bronze, Silver, Gold that provide services from different levels of storage pools based on the paid fees. Various combinations of storage options can be provided to containers as catalogs for example.

When the simulation needs modification of the base image and then changes the importance or priority of the image, containers can request an appropriate level of storage service to SDS-based storage infrastructure in the form of metadata or policies.

6 Concluding Remarks

The hydroclimatic experiments require intensive computing powers and data storage because of the complexity of simulations and the big size of data to process. High-performance computing (HPC) technology has supported the hydrology and climatology scientists making their experiments feasible by providing parallel processing computing. However, current HPC built on hardware-based virtual machines does not support easy deployment and software reusability as well as high scalability due to its heavy weight. The new OS-based virtualization technique called container has been used by industry and various academic organizations. In this paper, we introduced hydroclimatic experiments and their fast deployment and high portability and scalability in container.

Besides containers, we also introduced software-defined storage technologies in connection with hydroclimatic simulations. Storage management issue for big data that the hydroclimatology researchers are facing can be addressed by adopting the software-defined storage technology. However, when hydroclimatology simulations are performed in the containerized environment, the SDS should be integrated with the containers in a way that can support containers. In this paper, we discussed that SDS supports containers by providing high flexibility, high scalability, and simple mobility as well as storage platform options to choose appropriate level of service quality.

References

1. Community earth system model (CESM). http://www.cesm.ucar.edu/
2. Koster RD et al (2004) Regions of strong coupling between soil moisture and precipitation. Science 305(5687):1138–1140
3. Deser C, Phillips AS, Bourdette V, Teng H (2012) Uncertainty in climate change projections: the role of internal variability. Climate Dyn 38:527–546. doi:10.1007/s00382-010-0977-x
4. Earth system grid. http://www.earthsystemsgrid.org/
5. Hill C, DeLuca C, Balaji V, Suarez M, da Silva A (2004) Architecture of the earth system modeling framework. Comput Sci Eng 6(1):18–28

6. Dunlap R, Mark L, Rugaber S, Balaji V, Chastang J, Cinquini L, DeLuca C, Middleton D, Murphy S (2008) Earth system curator: metadata infrastructure for climate modeling. Earth Sci Inf 1(3–4):131–149
7. Metaphore. http://metaforclimate.eu/
8. Basumallik A, Zhao L, Song XC, Sriver RL, Huber M (2007) A community climate system modeling portal for the TeraGrid. In: TeraGrid 2007 conference, Madison, 4–8 June
9. Zhao L, Lee W, Song CX, Huber M, Goldner A (2010) Bringing high performance climate modeling into the classroom. In: TeraGrid 2010 conference, Pittsburgh
10. iRODS: data grids, digital libraries, persistent archives, and real-time data systems. http://www.irods.org/
11. CoreOS. https://coreos.com/
12. p Docker. http://www.docker.com/
13. StackEngine survey. http://sdtimes.com/survey-70-enterprises-adopting-docker-containers/
14. Diffenbaugh NS, Pal JS, Trapp RJ, Giorgi F (2005) Fine-scale processes regulate the response of extreme events to global climate change. Proc Natl Acad Sci USA 102(44):15774–15778
15. Diffenbaugh NS, Pal JS, Giorgi F, Gao XJ (2007) Heat stress intensification in the Mediterranean climate change hotspot. Geophys Res Lett 34(11):L11706
16. Dominguez F, Kumar P, Vivoni ER (2008) Precipitation recycling variability and ecoclimatological stability–a study using NARR data. Part II: North American Monsoon Region. J Climate 21(20):5187–5203
17. Findell KL, Eltahir EAB (2003) Atmospheric controls on soil moisture-boundary layer interactions. Part I: framework development. J Hydrometeorol 4(3):552–569
18. Fischer EM, Seneviratne SI, Luthi D, Schar C (2007) Contribution of land-atmosphere coupling to recent European summer heat waves. Geophys Res Lett 34(6):L06707
19. Hoerling M, Eischeid J, Kumar A, Leung R, Mariotti A, Mo K, Schubert S, Seager R (2014) Causes and predictability of the 2012 at plains drought. Bull Am Meteorol Soc 95(2):269–282
20. Seager R, Hoerling M, Schubert S, Wang H, Lyon B, Kumar A, Nakamura J, Henderson N (2015) Causes of the 2011 to 2014 California drought. J Climate 28(18):150904104833007
21. Trenberth KE, Fasullo JT, Shepherd TG (2015) Attribution of climate extreme events. Nat Climate Change 5(8):725–730
22. Williams AP, Seager R, Abatzoglou JT, Cook BI, Smerdon JE, Cook ER (2015) Contribution of anthropogenic warming to California drought during 2012–2014. Geophys Res Lett 42(16):6819–6828
23. Seneviratne SI et al (2013) Impact of soil moisture-climate feedbacks on CMIP5 projections: first results from the GLACE-CMIP5 experiment. Geophys Res Lett 40(19):5212–5217
24. Hurrell JW et al (2013) The community earth system model: a framework for collaborative research. Bull Amer Meteor Soc 94:1339–1360. doi:10.1175/BAMS-D-12-00121.1
25. Kay JE et al (2015) The community earth system model (CESM) Large ensemble project: a community resource for studying climate change in the presence of internal climate variability. Bull Am Met Soc 96:1333–1349. doi:10.1175/BAMS-D-13-00255.1
26. What is software-defined atorage? http://headintotheclouds.com/what-is-software-defined-storage/
27. Carlson M, Yoder A, Schoeb L, Deel D, Pratt C, Lionetti C, Voigt D (2015) Software defined storage. Storage Netw Ind Assoc (SNIA) Tech Rep Jan 2015. http://www.snia.org/sds
28. Darabseh A, Al-Ayyoub M, Jararweh Y, Benkhelifa E, Vouk M, Rindos A (2015) SDStorage: a software defined storage experimental framework. In: Proceedings of the 2015 IEEE international conference on cloud engineering, IC2E'15, Tempe, pp 341–346
29. The UberCloud marketplace with samples of containerized application solutions. https://www.TheUberCloud.com/Marketplace
30. Adufu T, Choi J, Kim Y (2016) Is container-based technology a winner for high performance scientific applications? In: Network operations and management symposium, APNOMS 2015, Busan, pp 507–510
31. Bernstein D (2014) Containers and cloud: from LXC to Docker to kubernetes. IEEE Cloud Comput 1(3):81–84
32. Turnbull J (2014) The Docker book. http://www.dockerbook.com/
33. Docker-Hub. https://hub.docker.com/
34. Scale testing Docker Swarm to 30,000 containers. https://blog.docker.com/2015/11/scale-testing-docker-swarm-30000-containers/