A system, method and computer-readable storage devices for providing a climate data persistence service. A system configured to provide the service can include a climate data server that performs data and metadata storage and management functions for climate data objects, a compute-storage platform that provides the resources needed to support a climate data server, provisioning software that allows climate data server instances to be deployed as virtual climate data servers in a cloud computing environment, and a service interface, wherein persistence service capabilities are invoked by software applications running on a client device. The climate data objects can be in various formats, such as International Organization for Standards (ISO) Open Archival Information System (OAIS) Reference Model Submission Information Packages, Archive Information Packages, and Dissemination Information Packages. The climate data server can enable scalable, federated storage, management, discovery, and access, and can be tailored for particular use cases.
(51) **Int. Cl.**

- **G06F 17/30** (2006.01)
- **G06Q 50/26** (2012.01)

(58) **Field of Classification Search**

USPC ................. 707/661, 662, 668; 715/772, 835

See application file for complete search history.

(56) **References Cited**

**U.S. PATENT DOCUMENTS**

- 7,856,228 B2 * 12/2010 Lekutai ................. G01W 1/02
  340/601
- 2013/0014046 A1 * 1/2013 Watts ..................... G01W 1/00
  715/772
  709/204

* cited by examiner
FIG. 1
NetCDF KIT

UTILITIES LIBRARY

* ___________
* ___________
* ___________

SERVICES LIBRARY

INGEST
Put()

QUERY
GetFileNameByAttribute()

DOWNLOAD
Get()
GetFileNameByAttribute()

EXECUTE
AddMetadataByName()
DeleteObject()

STATUS
CheckStatus()

FIG. 3
FEDERATED DATA GRID

VM

COMPUTE-STOREAGE PLATFORM

FIG. 4
PERFORMING DATA AND METADATA STORAGE AND MANAGEMENT FUNCTIONS FOR CLIMATE DATA OBJECTS

PROVIDING THE RESOURCES NEEDED TO SUPPORT A CLIMATE DATA SERVER

PROVISIONING SOFTWARE THAT ALLOWS CLIMATE DATA SERVER INSTANCES TO BE DEPLOYED AS VIRTUAL CLIMATE DATA SERVERS IN A CLOUD COMPUTING ENVIRONMENT

INVOKING, VIA A SERVICE INTERFACE, PERSISTENCE SERVICE CAPABILITIES BY SOFTWARE APPLICATIONS RUNNING ON A CLIENT DEVICE

FIG. 6
SYSTEM AND METHOD FOR PROVIDING A CLIMATE DATA PERSISTENCE SERVICE

BACKGROUND

1. Technical Field
The present disclosure relates to climate data services and more specifically to a particular combination of technologies that can deliver climate data persistence as a service for large climate datasets.

2. Introduction
Climate models generate data that are of great value to society. Climate model outputs include retrospective analyses that model the historical state of the climate, estimates of current climate conditions, and projections of future climate conditions. Climate data analytics as a service provides an approach that makes it easier to access the data and perform data analyses where the data are stored before moving reduced, more usable products to the end user for further study. The current technologies are deficient, however, because no effective means exists for storing and managing data products that are dynamically created by climate data analytic systems. What is needed is an improved approach that makes it easier to store and manage specialized collections of climate data and the dynamically created products produced by climate data analytics.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 illustrates an example persistence service; FIG. 2 illustrates an example climate data server; FIG. 3 illustrates an example NetCDF kit; FIG. 4 illustrates an example data grid including a collection of federated virtual climate data servers; FIG. 5 illustrates an exemplary use of a persistence service; FIG. 6 illustrates an example method embodiment; and FIG. 7 illustrates an example system embodiment.

DETAILED DESCRIPTION

A system, method and computer-readable storage devices are disclosed which deliver climate data persistence as a service. The persistence service (PS) described here is an example embodiment of a key component of an example climate data analytics system. Various modifications and changes may be made to this embodiment without departing from the broader spirit and scope of the disclosure. In particular, alternative data storage technologies may be used as the basis for such a service.

FIG. 1 is a diagram showing the overall organization of the example persistence service. The service 100 includes a climate data server (CDS) 101 that performs the data and metadata storage and management functions of the service, a compute-storage platform 102 that provides the resources needed to run the climate data server, virtualization and provisioning software 103 that allows multiple climate data server instances to be built as virtual machine images, and service interface 104 whereby the capabilities of the persistence service are made available to client devices.

FIG. 2 is a diagram showing an example climate data server (CDS) 200. CDS can implement the core functionality of the persistence service. The CDS can be a data server software appliance specialized to the needs of a managed collection of climate-related scientific data. CDS is designed to take advantage of the flexible resource allocation capabilities afforded by cloud computing. As described in greater detail below, to allow for ease of collections integration and support the full information lifecycle requirements of a scientific archive, CDS should be built around using scalable data grid technology.

The climate data server 200 used by the example persistence service includes the Integrated Rule-Oriented Data System (RODS) data grid software 201 running in a SLES 11 SP3 operating system environment 203. Provisioning software 103 can encapsulate the operating system and iRODS in a virtual machine image. Various application-specific kits 202 are then used to specialize the CDS's functionality for particular uses. The example CDS has a Network Common Data Format (NetCDF) kit 202.1, a GeoTIFF data management kit 202.2, and can include other kits 202.3 with different formats and capabilities. In this way, the system can provide a virtual climate data server configured with application-specific kits.

FIG. 3 illustrates an example NetCDF kit 300 that might be used in an embodiment of the climate data server. The NetCDF kit includes a services library 301 that contains the methods that implement the core functionality of the service and a utilities library 302 that provides ancillary methods that support the service functions. The system can organize the NetCDF kit's methods to contribute to the integrated analytics-archive management perspective of climate data analytics as a service. The functional capabilities of the service can correspond to the Open Archival Information System (OAIS) Reference Model data flow categories of an operational archive.

An OAIS is an archive associated with an organization of people and systems that have accepted the responsibility to preserve information and make that information available for a designated community. The term OAIS also refers to the International Standards Organization (ISO) OAIS Reference Model for an OAIS. While the OAIS model typically refers to space agencies, the OAIS model can be useful in a wide variety of other organizations and institutions with digital archiving needs. OAIS provides a framework for the understanding and increased awareness of archival concepts needed for long-term digital information preservation and access and provides the concepts needed by non-archival organizations to be effective participants in the preservation process.

The OAIS-based capability categories used to organize a persistence service's methods are ingest, query, order, download, execute, and status. "Ingest" methods input objects into the system, "query" methods retrieve metadata relating to data objects in the service, "order" methods dynamically create data objects, and "download" methods retrieve objects from the service. The execute and status categories have been added to the OAIS functional model to accommodate the dynamic nature of a climate data analytics as a service-type archive. "Execute" methods initiate service-definable operations that can extend the functionality of a service, and "status" methods check on the progress of running operations.

In one example embodiment, the NetCDF kit's services library contains five OAIS categories of capability: ingest, query, download, execute, and status, which collectively implement the classic "CRUD" operations of an archive: create, read, update, and delete data objects and metadata associated with the data objects. In further compliance with the OAIS Reference Model, the data objects managed by the NetCDF kit represent OAIS's Submission Information Package, Archive Information Package, and Dissemination Information Package abstractions. The NetCDF kit can treat the persistence service's data objects as OAIS packages.
In one example embodiment, the persistence service includes or invokes a Put order method that stores a user-specified payload and returns to the calling application a session identifier, a one word status update, and a detailed description of the session. A GetFileNameByNameAttribute query method can perform a metadata search operation on the data objects stored in the service according to user-specified parameters. The service can include two download capabilities: a Get method to download a previously ingested data object according to user-specified parameters, and a GetFileNameByNameAttribute method to query for a data object name according to user-specified parameters. In both cases, the user-specified parameters include a file name for a data object to be operated upon, a target destination path for an output data object, and an optional overwrite specification that stipulates whether the downloaded object destructively writes over an existing data object or version of the object.

An AddMetaDataByName execute method can add metadata to a stored data object according to user-specified parameters including a file name for the target data object, a name specification for the metadata key to be associated with the target data object, a corresponding value for the key, and an optional unit specification. Outputs from the operation can include a unique session ID, a one word status message, and a detailed description of the session. A second execute capability, a DeleteObject method, can remove a stored data object for the service and returns a status message. Finally, a CheckStatus status method can check on the progress of service request based on an input session ID and return a status update message. User-specified parameters of the DeleteObject method can include a file name for the target data object and output can be a status update.

In one example embodiment, the NetCDF kit’s utilities library contains methods that enable the persistence service to manage metadata in accordance with the OAIS Reference Model’s metadata taxonomy, which recognizes four categories of metadata: Representation Information, Preservation Description Information, Policy Information, and Discovered Metadata. Specifically, the NetCDF kit’s utilities library can contain a method that extracts the Representation Information and Preservation Description Information embedded in the self-describing NetCDF files and stores that information as a set of internal iRODS database tables managed by method libraries. This externalized metadata can facilitate discovery, and allow clients to perform searches over the NetCDF data objects stored in the persistence service without opening files to access embedded metadata. The utilities library can also contain a method that logs object-level actions, thereby enabling low-level system performance monitoring and optimization.

FIG. 4 illustrates an example federated data grid comprising a collection of federated virtual climate data servers (vCDS) configured to practice the method. The steps outlined herein are exemplary and can be implemented in any combination thereof, including combinations that exclude, add, or modify certain steps.

Having disclosed some basic system components and concepts, the disclosure now turns to the exemplary method embodiment shown in FIG. 6. For the sake of clarity, the method is described in terms of an exemplary system as shown in FIG. 6 configured to practice the method. The steps outlined herein are exemplary and can be implemented in any combination thereof, including combinations that exclude, add, or modify certain steps.

A system configured according to this disclosure can provide a climate data persistence service. The data persistence service can include a climate data server that performs data and metadata storage and management functions according to user-specified parameters. The climate data server can be an International Organization for Standards (ISO) Open Archival Information System (OAIS) Reference Model Submission Information Packages, Archive Information Packages, and Dissemination Information Packages.

The data persistence service can include a compute-storage platform that provides the resources needed to support a climate data server. The data persistence service can further include provisioning software that allows data server instances to be deployed as virtual climate data servers in a cloud computing environment. The data persistence service can include a service interface, wherein persistence service capabilities are invoked by software applications running on a client device.
for the target session. The various capabilities can collectively implement the classic create, read, update, and delete operations of an operational archive. The provisioning software can be RPM Package Manager scripts that build instances of the climate data server software stack in a cloud computing environment.

The services interface can include an adapter module that maps service requests from external client software applications to specific capabilities of the persistence service, and a representational state transfer server module that communicatively links the persistence service to external client software applications. The adapter module can include a service request-capability mapping based on the International Organization for Standards Archival Information System Reference Model data flow categories. In the representational state transfer module, the persistence service communicates to external client software applications through International Organization for Standards Open Archival Information System Reference Model-based uniform resource locators including an ingest endpoint of the general form http://<base URL>/ingest.php?service=<service_name>&request=<operation>&parameters=<parameters>, a download endpoint of the general form http://<base URL>/download.php?service=<service_name>&request=<operation>&parameters=<parameters>, an execute endpoint of the general form http://<base URL>/execute.php?service=<service_name>&request=<operation>&parameters=<parameters>, and a status endpoint of the general form http://<base URL>/status.php?service=<service_name>&request=<operation>&parameters=<parameters>. The services interface can be implemented as a PHP program.

Various embodiments of the disclosure are described in detail below. While specific implementations are described, it should be understood that this is done for illustration purposes only. Other components and configurations may be used without parting from the spirit and scope of the disclosure.

With reference to FIG. 7, an exemplary system and/or computing device 700 includes a processing unit (CPU or processor) 720 and a system bus 710 that couples various system components including the system memory 730 such as read only memory (ROM) 740 and random access memory (RAM) 750 to the processor 720. The system 700 can include a cache 722 of high-speed memory connected directly with, in close proximity to, or integrated as part of the processor 720. The system 700 copies data from the memory 730 and/or the storage device 760 to the cache 722 for quick access by the processor 720. In this way, the cache provides a performance boost that avoids processor 720 delays while waiting for data. These and other modules can control or be configured to control the processor 720 to perform various operations or actions. Other system memory 730 may be available for use as well. The memory 730 can include multiple different types of memory with different performance characteristics. It can be appreciated that the disclosure may operate on a computing device 700 with more than one processor 720 or on a group or cluster of computing devices networked together to provide greater processing capability. The processor 720 can include any general purpose processor and a hardware module or software module, such as module 1 762, module 2 764, and...
In another aspect, the system can use a processor and/or digital signal processor (DSP) hardware, read-only memory (ROM) 740 for storing software performing the operations described below, and random access memory (RAM) 750 for storing results. Very large scale integration (VLSI) hardware embodiments, as well as custom VLSI circuitry in combination with a general purpose DSP circuit, may also be provided.

The logical operations of the various embodiments are implemented as: (1) a sequence of computer implemented steps, operations, or procedures running on a programmable circuit within a general use computer, (2) a sequence of computer implemented steps, operations, or procedures running on a specific-use programmable circuit; and/or (3) interconnected machine modules or program engines within the programmable circuits. The system 700 shown in FIG. 7 can practice all or part of the recited methods, can be a part of the recited systems, and/or can operate according to instructions in the recited tangible computer-readable storage devices. Such logical operations can be implemented as modules configured to control the processor 720 to perform particular functions according to the programming of the module. For example, FIG. 7 illustrates three modules Mod1 762, Mod2 764 and Mod3 766 which are modules configured to control the processor 720. These modules may be stored on the storage device 760 and loaded into RAM 750 or memory 730 at runtime or may be stored in other computer-readable memory locations.

One or more parts of the example computing device 700, up to and including the entire computing device 700, can be virtualized. For example, a virtual processor can be a software object that executes according to a particular instruction set, even when a physical processor of the same type as the virtual processor is unavailable. A virtualization layer or a virtual “host” can enable virtualized components of one or more different computing devices or device types by translating virtualized operations to actual operations.
Ultimately, however, virtualized hardware of every type is implemented or executed by some underlying physical hardware. Thus, a virtualization compute layer can operate on top of a physical compute layer. The virtualization compute layer can include one or more of a virtual machine, an overlay network, a hypervisor, virtual switching, and any other virtualization application.

The processor 720 can include all types of processors disclosed herein, including a virtual processor. However, when referring to a virtual processor, the processor 720 includes the software components associated with executing the virtual processor in a virtualization layer and underlying hardware necessary to execute the virtualization layer. The system 700 can include a physical or virtual processor 720 that receive instructions stored in a computer-readable storage device, which cause the processor 720 to perform certain operations. When referring to a virtual processor 720, the system also includes the underlying physical hardware executing the virtual processor 720.

Embodiments within the scope of the present disclosure may also include tangible and/or non-transitory computer-readable storage devices for carrying or having computer-executable instructions or data structures stored thereon. Such tangible computer-readable storage devices can be any available device that can be accessed by a general purpose or special purpose computer, including the functional design of any special purpose processor as described above. By way of example, and not limitation, such tangible computer-readable devices can include RAM, ROM, EEPROM, CD-ROM or other optical disk storage, magnetic disk storage, other magnetic storage devices, or any other device which can be used to carry or store desired program code in the form of computer-executable instructions, data structures, or processor states. When information or instructions are provided via a network or another communications connection (either hardwired, wireless, or combination thereof) to a computer, the computer properly views the connection as a computer-readable medium. Thus, any such connection is properly termed a computer-readable medium. Combinations of the above should also be included within the scope of the computer-readable storage devices.

Computer-executable instructions include, for example, instructions and data which cause a general purpose computer, special purpose computer, or special purpose processing device to perform a certain function or group of functions. Computer-executable instructions also include program modules that are executed by computers in standalone or network environments. Generally, program modules include routines, programs, components, data structures, objects, and the functions inherent in the design of special-purpose processors, etc. that perform particular tasks or implement particular abstract data types. Computer-executable instructions, associated data structures, and program modules represent examples of the program code means for executing steps of the methods disclosed herein. The particular sequence of such executable instructions or associated data structures represents examples of corresponding acts for implementing the functions described in such steps.

Other embodiments of the disclosure may be practiced in network computing environments with many types of computer system configurations, including personal computers, handheld devices, multi-processor systems, microprocessor-based or programmable consumer electronics, network PCs, minicomputers, mainframe computers, and the like. Embodiments may also be practiced in distributed computing environments where tasks are performed by local and remote processing devices that are linked (either by hardwired links, wireless links, or by a combination thereof) through a communications network. In a distributed computing environment, program modules may be located in both local and remote memory storage devices.

The various embodiments described above are provided by way of illustration only and should not be construed to limit the scope of the disclosure. For example, the principles herein apply generally to climate data, but can also apply to other non-climate data sets. Various modifications and changes may be made to the principles described herein without following the example embodiments and applications illustrated and described herein, and without departing from the spirit and scope of the disclosure. Claim language reciting "at least one of" a set indicates that one member of the set or multiple members of the set satisfy the claim.

We claim:

1. A system for providing a climate data persistence service: 
   a. a climate data server that performs data and metadata storage and management functions for climate data objects; 
   b. a compute-storage platform that provides a number of resources needed to support a climate data server; 
   c. provisioning software that allows climate data server instances to be deployed as virtual climate data servers in a cloud computing environment; and 
   d. a service interface, wherein persistence service capabilities are invoked by software applications running on a client device; 

   wherein the climate data server further comprises: 
   a. a computer-readable storage device having stored therein data grid software which, when executed by a processor, causes the processor to enable scalable, federated storage, management, discovery, and access; 
   b. one or more application-specific kits that tailor the climate data server to particular uses and to particular file formats; 
   c. wherein at least one of the one or more application-specific kits comprises a Network Common Data Format kit; wherein the Network Common Data Format kit comprises: 
      a. a services library, wherein a plurality of software applications implement the capabilities of the service; and 
      b. a utilities library, wherein a plurality of software applications implement the support functions of the service; 
   d. wherein the utilities library further comprises: 
      a. a first function that extracts the embedded metadata in Network Common Data Format files and stores and manages the metadata separate from the stored data objects; 
      b. a second function that builds, populates, and manages the internal database tables of data grid software according to International Organization for Standards Open Archival Information System Reference Model metadata categories of Representation Information, Preservation Description Information, Policy Information, and Discovered Metadata; and 
      c. a third function that logs object-level actions within the data grid software; and 
   e. an operating system that enables the climate data server to run on the compute-storage platform.

2. The system of claim 1, wherein the climate data objects represent at least one of International Organization for Standards (ISO) Open Archival Information System (OAIS)
3. The system of claim 1, wherein the data grid software comprises an open source enterprise-ready distribution of the Integrated Rule-Oriented Data System Version 4.0 data management system.

4. The system of claim 1, wherein the operating system is SLES 11, SP 3.

5. The system of claim 1, wherein the services library and the utilities library comprise Python classes.

6. The system of claim 1, wherein the services library comprises methods that implement capabilities of the service that correspond to the International Organization for Standards Open Archival Information System Reference Model data flow categories of an operational archive comprising:

   ingest capabilities that input data objects to the service;
   query capabilities that retrieve metadata relating to data objects in the service;
   order capabilities that dynamically create data objects in the service;
   download capabilities that retrieve data objects from a service;
   execute capabilities that initiate service-definable operations; and
   status capabilities that check on the progress of an order operation.

7. The system of claim 6, wherein the ingest capabilities further comprise a Put method that stores a user-specified input payload in the persistence service.

8. The system of claim 7, wherein outputs from the Put method comprise a unique session identifier for an ingest session, a one word status update, and a detailed description of the ingest session.

9. The system of claim 6, wherein the query capabilities further comprise a GetFileNameByAttribute method that performs a metadata search operation on the data objects stored in the persistence service according to user-specified selection parameters.