Scibox: Online Sharing of Scientific Data via the Cloud

Jian Huang[†], Xuechen Zhang[†], Greg Eisenhauer[†], Karsten Schwan[†]
Matthew Wolf[†,‡], Stephane Ethier[‡], Scott Klasky[‡]

[†]CERCS Research Center, Georgia Tech
[‡]Princeton Plasma Physics Laboratory
[‡]Oak Ridge National Laboratory

Supported in part by funding from the US Department of Energy for DOE SDAV SciDac

Outline

- Background and Motivation
- Problems and Challenges
- Design and Implementation
- Evaluation
- Conclusion and Future Work

Cloud Storage is Popular

Easy-of-use

Pay-as-you-go model

Universal accessibility

Good scalability and durability

Cloud Storage is Popular

Easy-of-use

Pay-as-you-go model

Universal accessibility

Good scalability and durability

Works based on cloud storage

- Dropbox, GoogleDrive, iCloud, SkyDrive, and etc.

Cloud Storage is Popular

Easy-of-use

Pay-as-you-go model

Universal accessibility

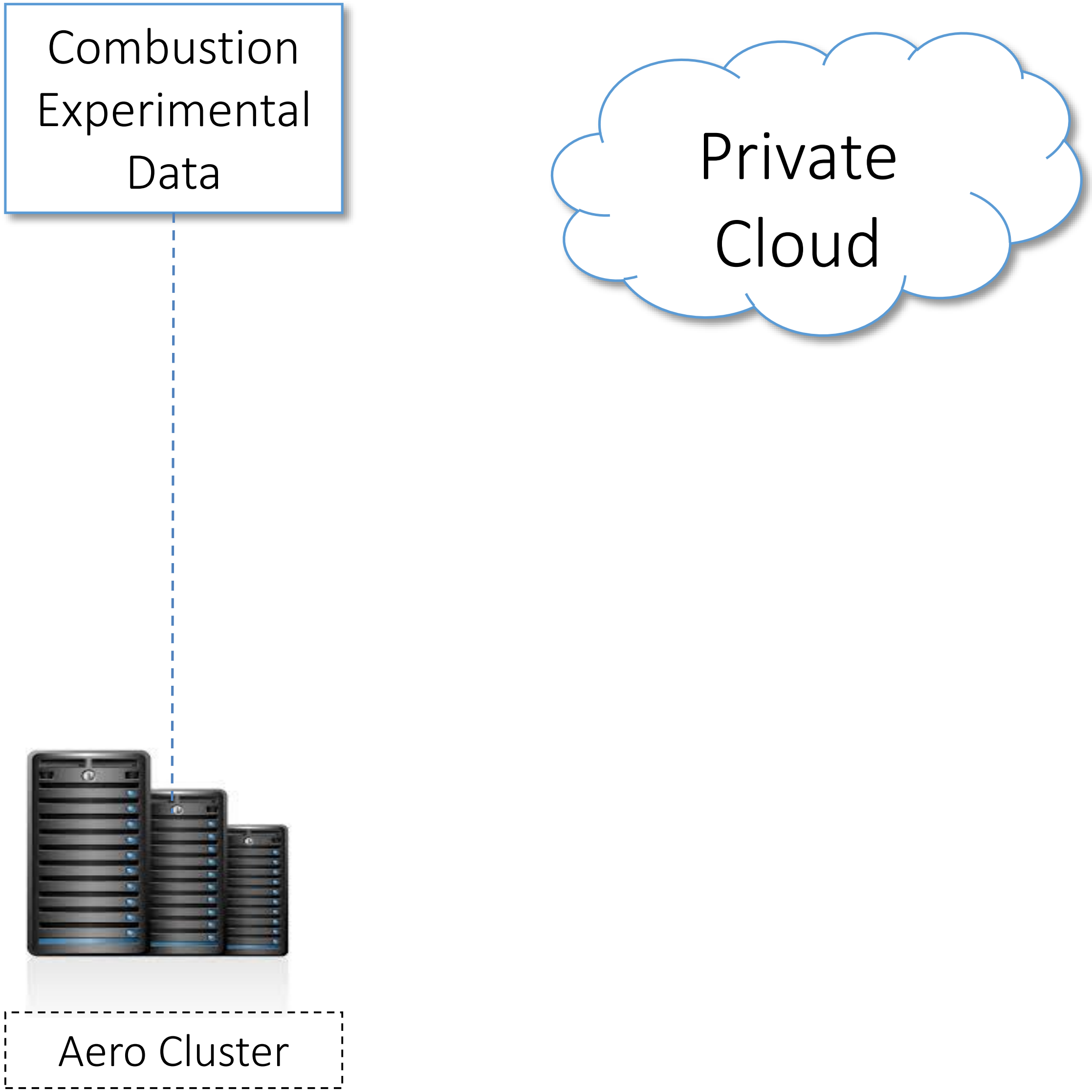
Good scalability and durability

Works based on cloud storage

- Dropbox, GoogleDrive, iCloud, SkyDrive, and etc.

→ Scibox: focus on scientific data sharing

Use Cases for Cloud Storage

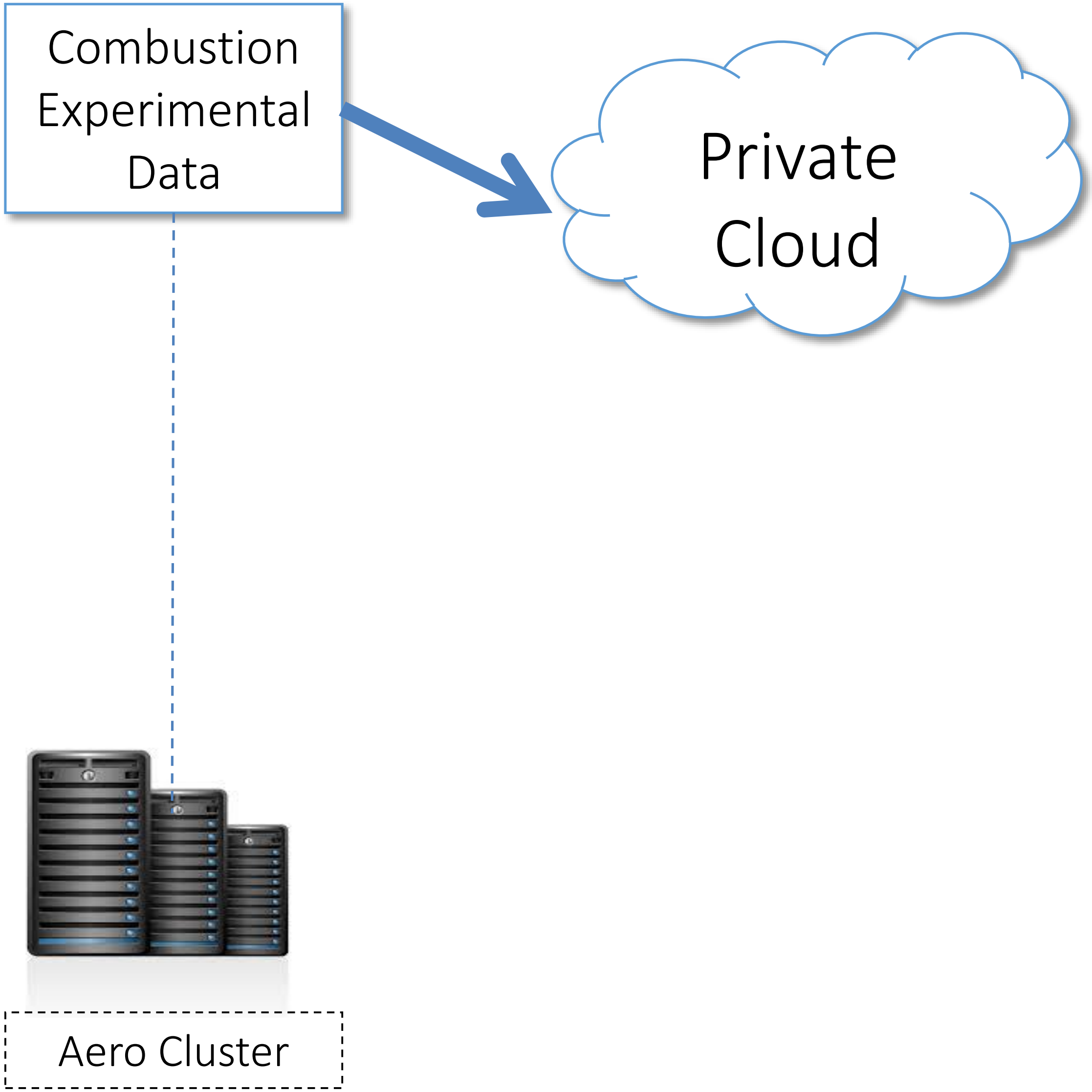


Combustion Experimental Data

Private Cloud

Aero Cluster

Use Cases for Cloud Storage

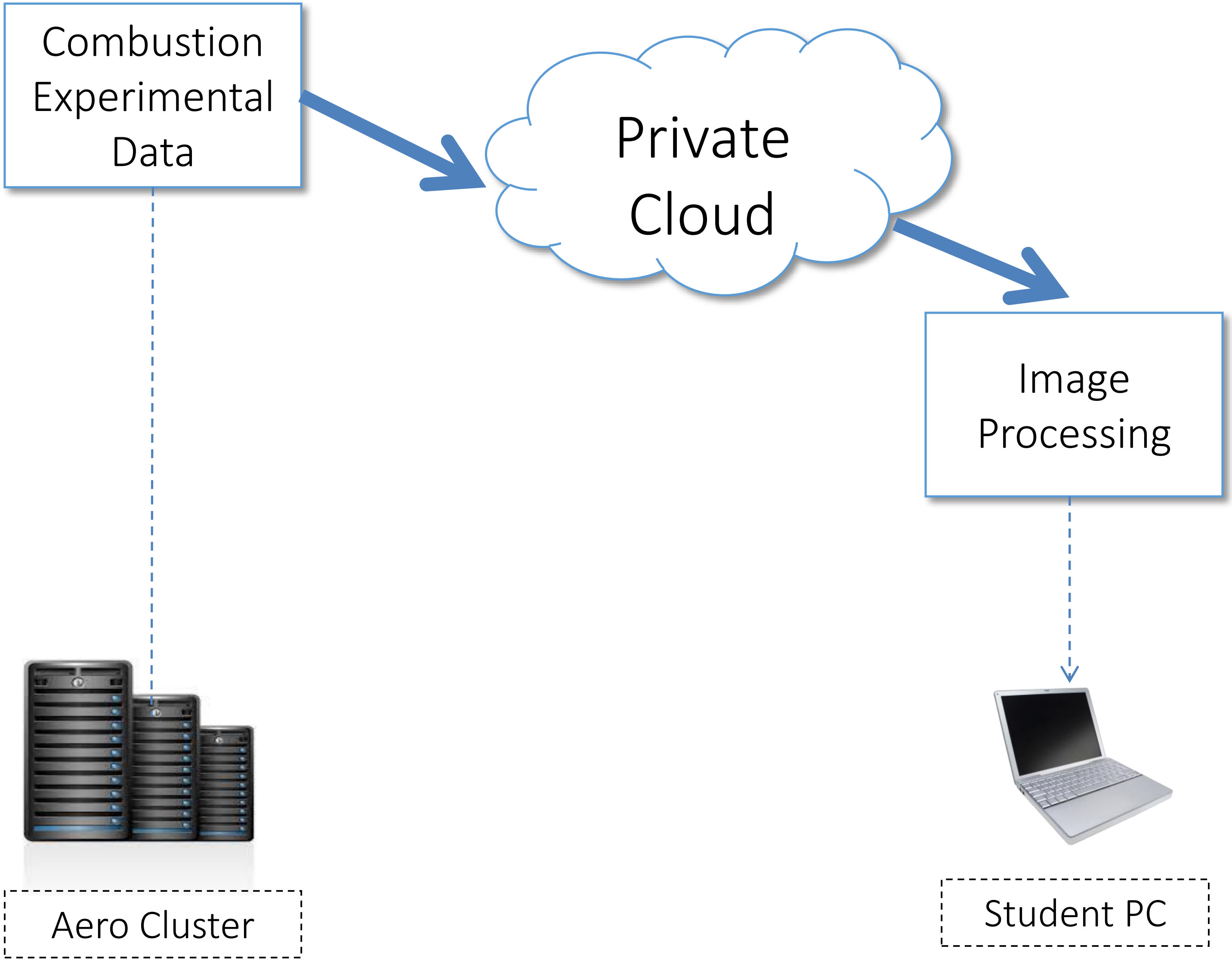


Combustion Experimental Data

Private Cloud

Aero Cluster

Use Cases for Cloud Storage



Combustion Experimental Data

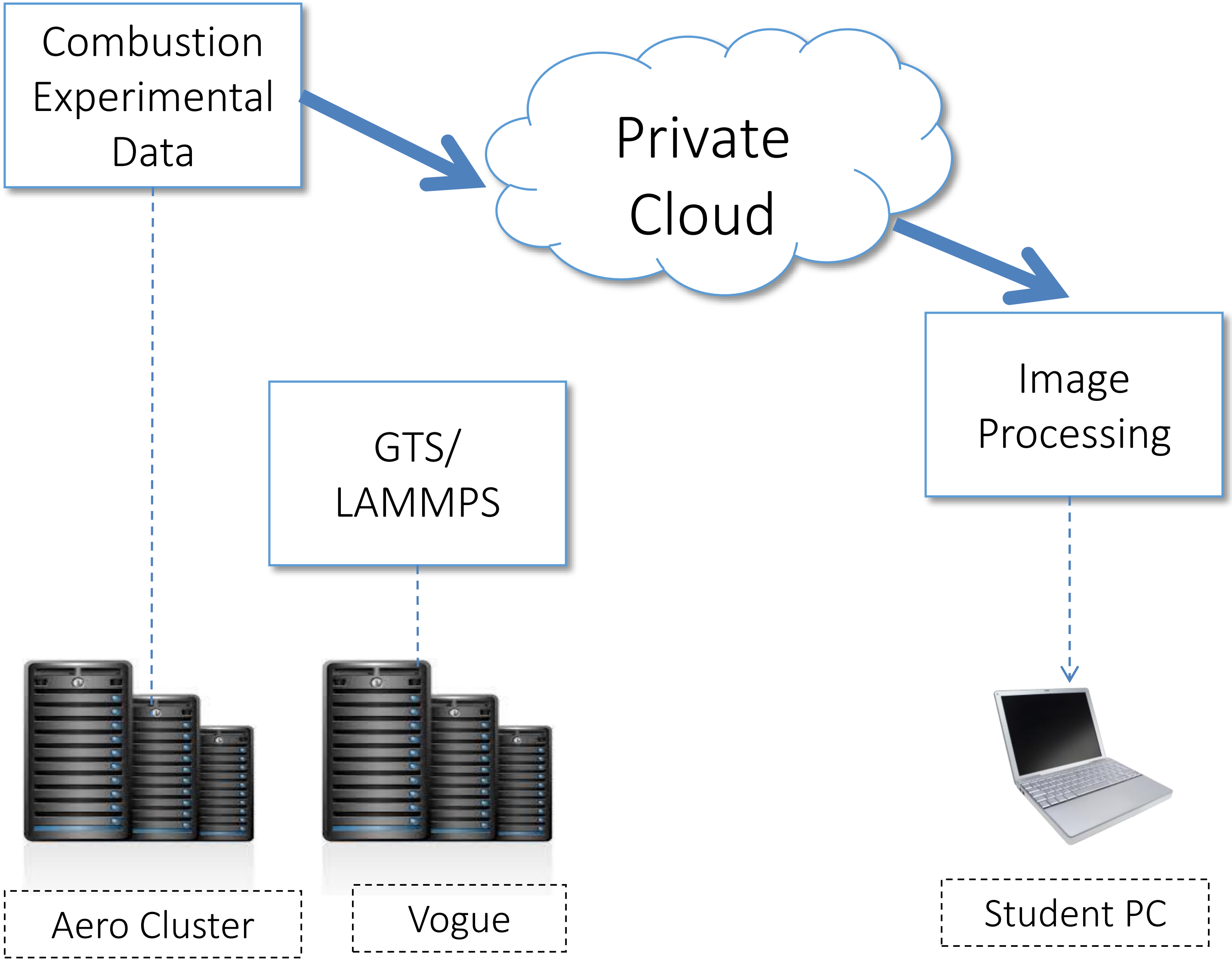
Private Cloud

Image Processing

Aero Cluster

Student PC

Use Cases for Cloud Storage



Combustion Experimental Data

Private Cloud

Image Processing

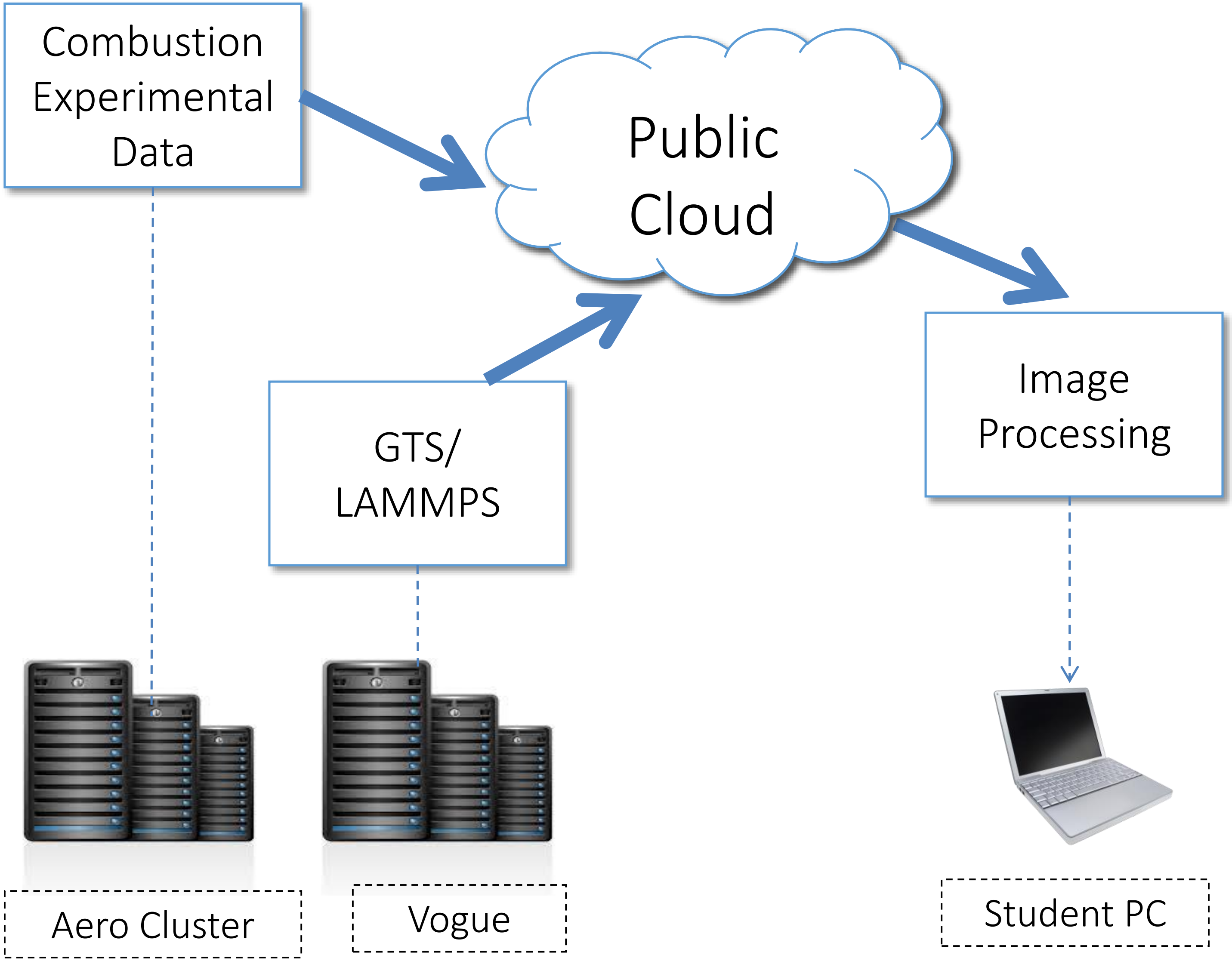
GTS/ LAMMPS

Aero Cluster

Vogue

Student PC

Use Cases for Cloud Storage



Combustion Experimental Data

Public Cloud

GTS/ LAMMPS

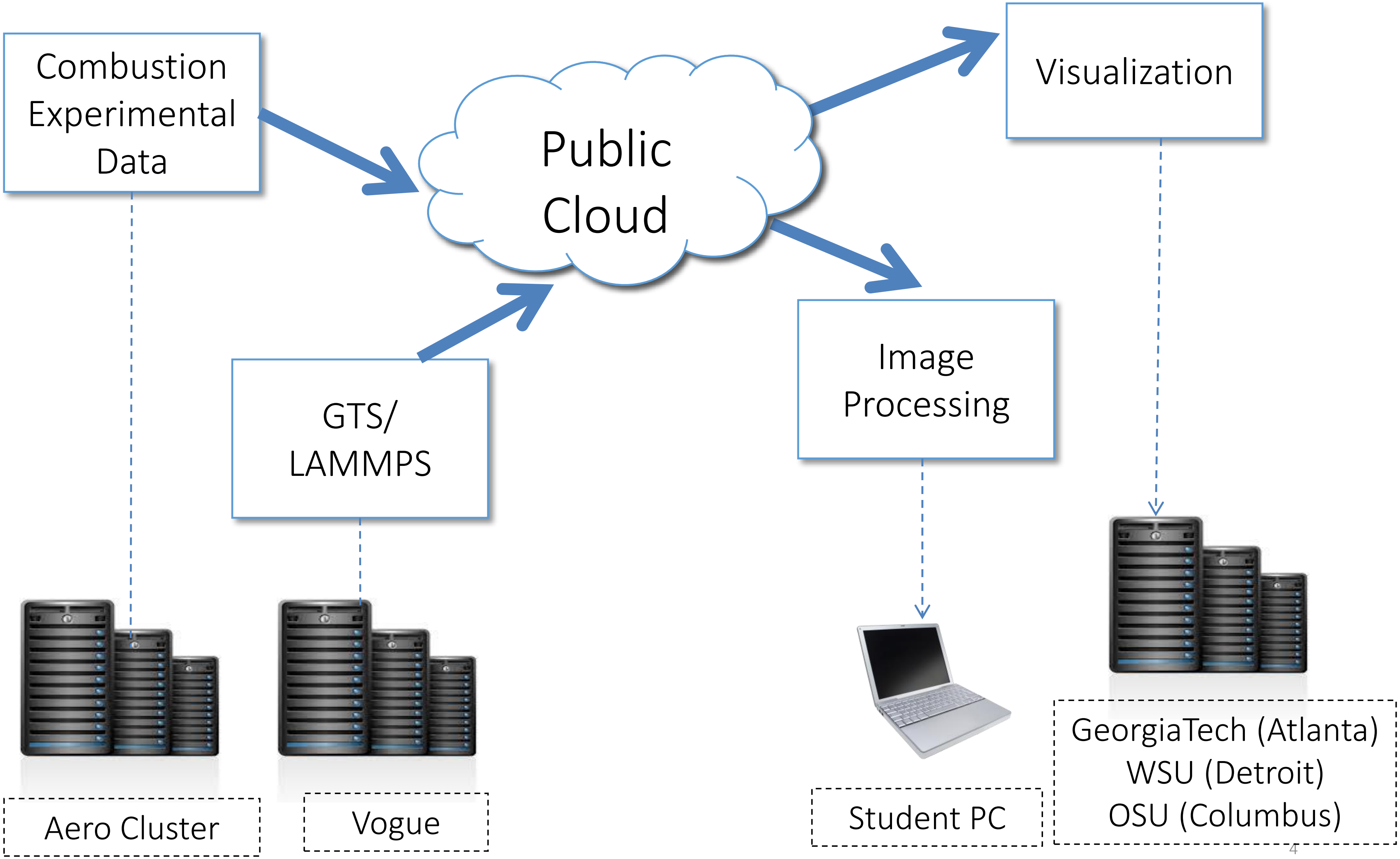
Image Processing

Aero Cluster

Vogue

Student PC

Use Cases for Cloud Storage



Combustion Experimental Data

Public Cloud

GTS/ LAMMPS

Visualization

Image Processing

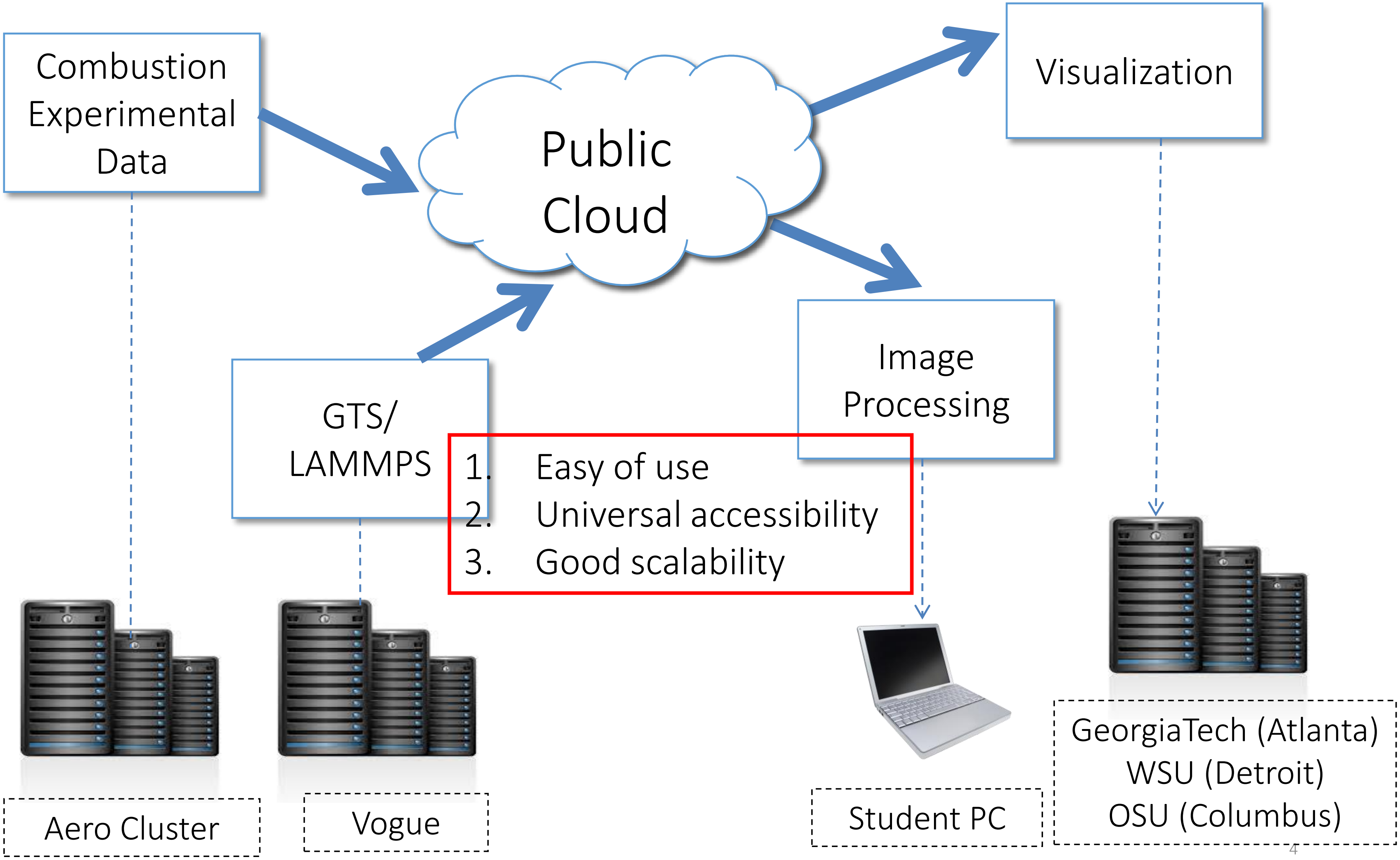
Aero Cluster

Vogue

Student PC

GeorgiaTech (Atlanta)
WSU (Detroit)
OSU (Columbus)

Use Cases for Cloud Storage



1. Easy of use
2. Universal accessibility
3. Good scalability

Outline

- Background and Motivation
- Problems and Challenges
- Design and Implementation
- Evaluation
- Conclusion and Future Work

Cloud Storage is Not Ready for HEC

Scientific applications are data-intensive

- Generate large amounts of data

Cloud Storage is Not Ready for HEC

Scientific applications are data-intensive

- Generate large amounts of data

Networking bandwidth is limited

- Inadequate levels of ingress and egress bandwidths available to/from remote cloud stores

Cloud Storage is Not Ready for HEC

Scientific applications are data-intensive

- Generate large amounts of data

Networking bandwidth is limited

- Inadequate levels of ingress and egress bandwidths available to/from remote cloud stores

High costs imposed by cloud providers

- Expensive for large amounts of data when using the pay-as-you-go model

Cloud Storage is Not Ready for HEC

Scientific applications are data-intensive

- Generate large amounts of data

Networking bandwidth is limited

- Inadequate levels of ingress and egress bandwidths available to/from remote cloud stores

High costs imposed by cloud providers

- Expensive for large amounts of data when using the pay-as-you-go model

An example:

A GTS runs on 29K cores on the Jaguar machine at OLCF generates over 54 Terabytes of data in a 24 hour period.

Amazon S3: ~$0.03/GB for storage and $0.09/GB for data transfer out.

Cost: $6635.52/day, increases with increasing number of collaborators

Problem: Too Much Data Movement

Issue: naïve approach transfers lots of data, even if only some of it is needed

Time step 0

A_0 A_1 A_2 … A_n

B_0 B_1 B_2 … B_n

C_0 C_1 C_2 … C_n

Time step 1

A_0 A_1 A_2 … A_n

B_0 B_1 B_2 … B_n

C_0 C_1 C_2 … C_n

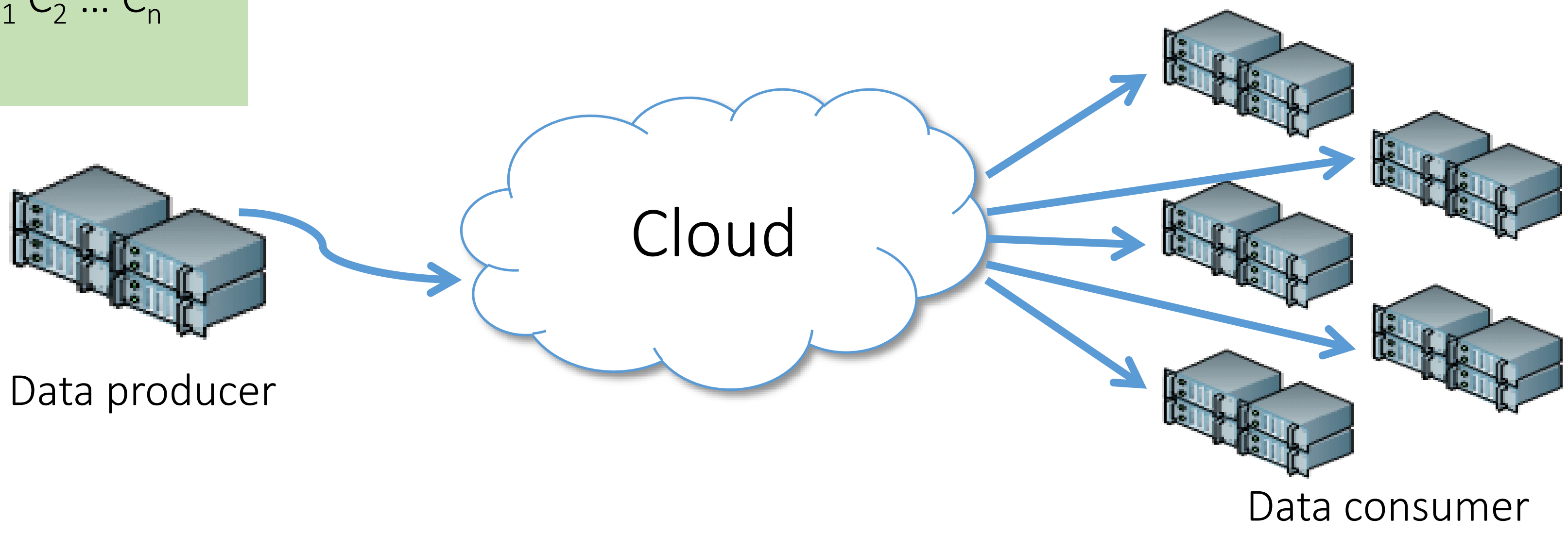
…

Example

Output of GTS fusion modeling simulation:

Checkpoint data, diagnosis data, visualization data and etc.

Each data subset includes many elements



Problem: Too Much Data Movement

Issue: naïve approach transfers lots of data, even if only some of it is needed

Time step 0

A_0 A_1 A_2 … A_n

B_0 B_1 B_2 … B_n

C_0 C_1 C_2 … C_n

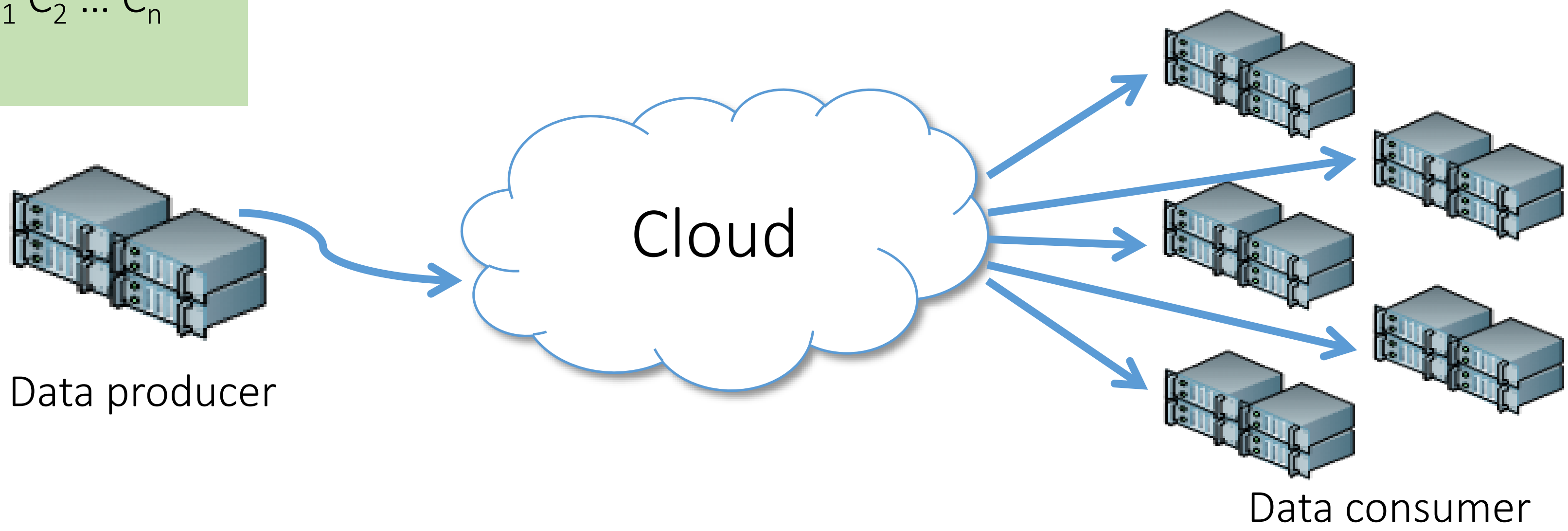
Time step 1

A_0 A_1 A_2 … A_n

B_0 B_1 B_2 … B_n

C_0 C_1 C_2 … C_n

…



Problem: Too Much Data Movement

Issue: naïve approach transfers lots of data, even if only some of it is needed

Time step 0

A_0 A_1 A_2 … A_n

B_0 B_1 B_2 … B_n

C_0 C_1 C_2 … C_n

Time step 1

A_0 A_1 A_2 … A_n

B_0 B_1 B_2 … B_n

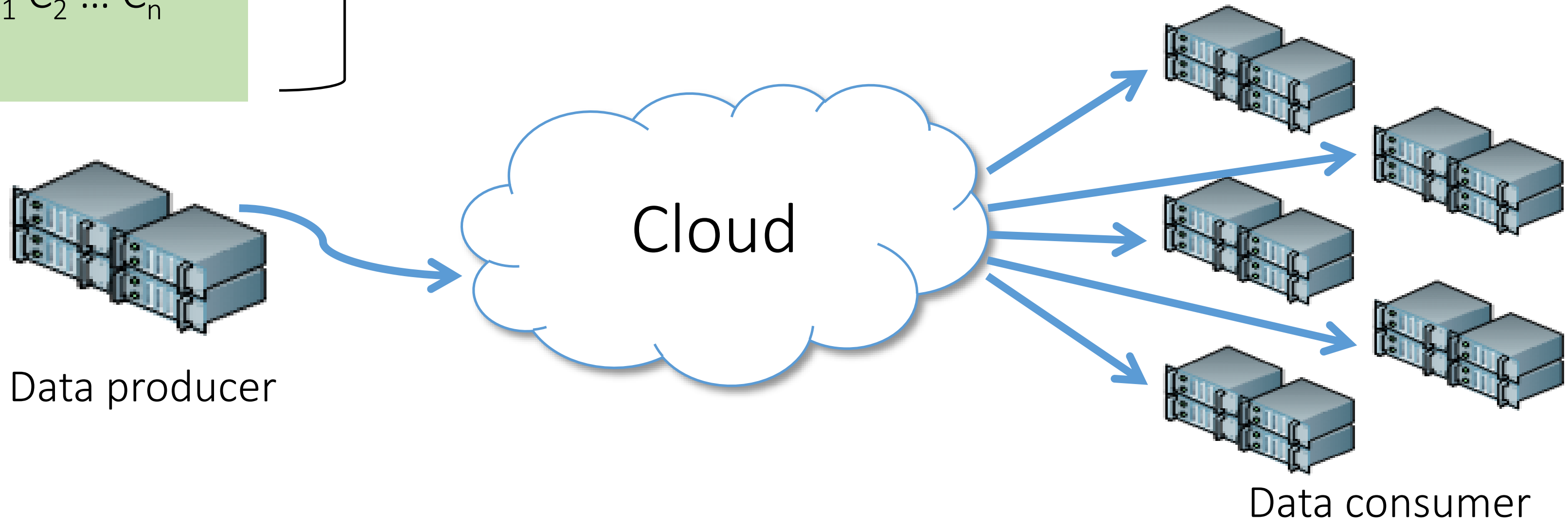
C_0 C_1 C_2 … C_n

…

- Scientific data formats: BP/HDF5

  Structured and meta-data rich
- Standard I/O interface: ADIOS

  Almost transparent to users

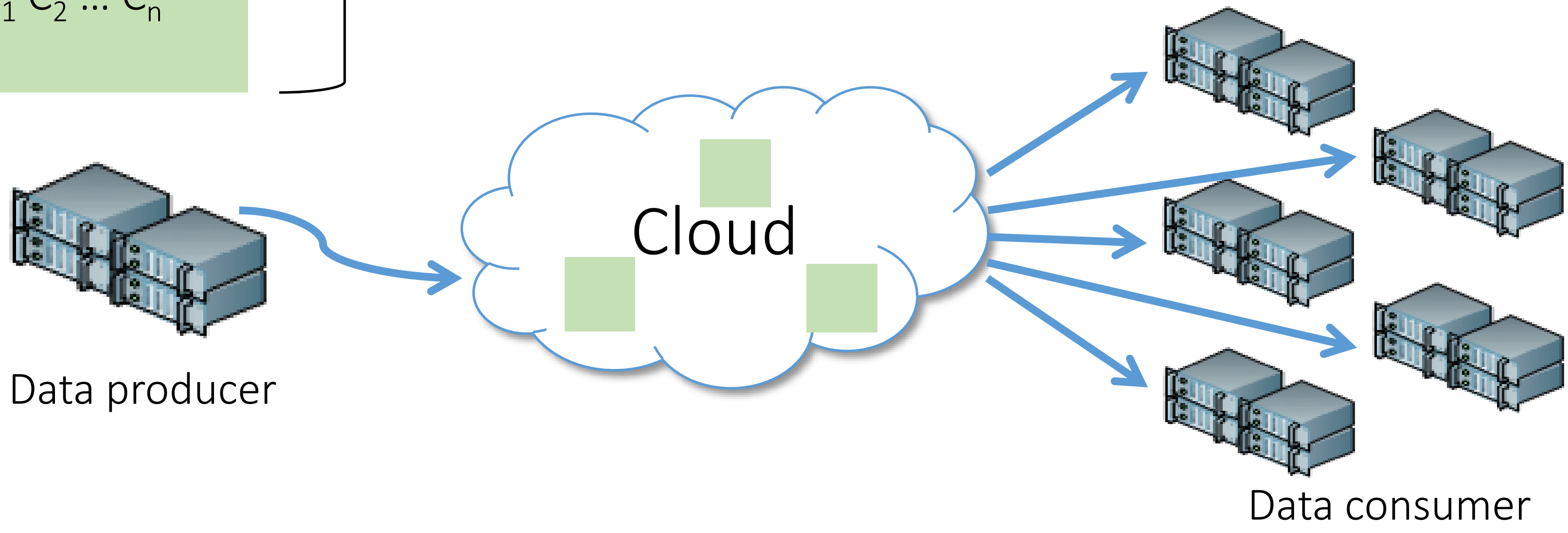


Problem: Too Much Data Movement

Issue: naïve approach transfers lots of data, even if only some of it is needed

Time step 0
A_0 A_1 A_2 … A_n
B_0 B_1 B_2 … B_n
C_0 C_1 C_2 … C_n
Time step 1
A_0 A_1 A_2 … A_n
B_0 B_1 B_2 … B_n
C_0 C_1 C_2 … C_n
…

- Scientific data formats: BP/HDF5
  Structured and meta-data rich
- Standard I/O interface: ADIOS
  Almost transparent to users



Problem: Too Much Data Movement

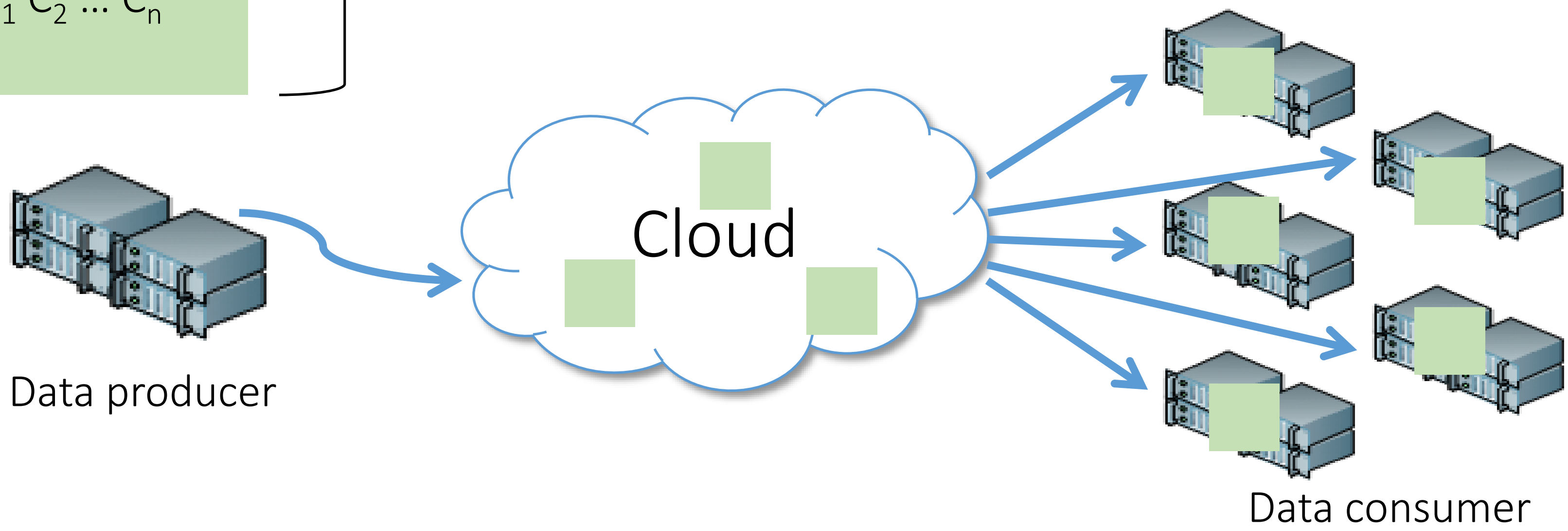
Issue: naïve approach transfers lots of data, even if only some of it is needed

Time step 0
A_0 A_1 A_2 … A_n
B_0 B_1 B_2 … B_n
C_0 C_1 C_2 … C_n
Time step 1
A_0 A_1 A_2 … A_n
B_0 B_1 B_2 … B_n
C_0 C_1 C_2 … C_n
…

- Scientific data formats: BP/HDF5

  Structured and meta-data rich
- Standard I/O interface: ADIOS

  Almost transparent to users



Problem: Too Much Data Movement

Issue: naïve approach transfers lots of data, even if only some of it is needed

Time step 0

A_0 A_1 A_2 … A_n

B_0 B_1 B_2 … B_n

C_0 C_1 C_2 … C_n

Time step 1

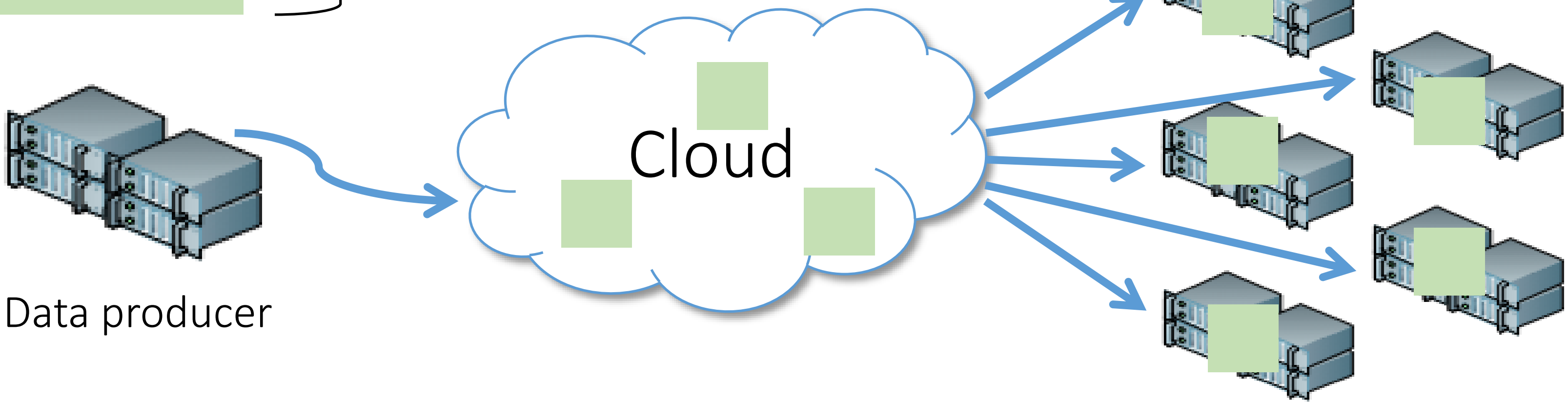
A_0 A_1 A_2 … A_n

B_0 B_1 B_2 … B_n

C_0 C_1 C_2 … C_n

…

- Scientific data formats: BP/HDF5
  Structured and meta-data rich
- Standard I/O interface: ADIOS
  Almost transparent to users



Data consumer

Goal: Reduce data transfer from producers to consumers

Solutions for Minimizing Data Transfers for Data Sharing

Compression

- Helps, but compression ratio can be low for floating-point scientific data

Solutions for Minimizing Data Transfers for Data Sharing

Compression

- Helps, but compression ratio can be low for floating-point scientific data

Data selection

- Files: users can ask for subsets of data files, by specifying file offsets
- Requires knowledge about data layout in files

Solutions for Minimizing Data Transfers for Data Sharing

Compression

- Helps, but compression ratio can be low for floating-point scientific data

Data selection

- Files: users can ask for subsets of data files, by specifying file offsets
- Requires knowledge about data layout in files

Content-based data indexing

- Useful, but may require large amounts of meta-data

Solutions for Minimizing Data Transfers for Data Sharing

Compression

- Helps, but compression ratio can be low for floating-point scientific data

Data selection

- Files: users can ask for subsets of data files, by specifying file offsets
- Requires knowledge about data layout in files

Content-based data indexing

- Useful, but may require large amounts of meta-data

Scibox Approaches:

- Filter unnecessary data at producer-side via metadata (uploads)

Solutions for Minimizing Data Transfers for Data Sharing

Compression

- Helps, but compression ratio can be low for floating-point scientific data

Data selection

- Files: users can ask for subsets of data files, by specifying file offsets
- Requires knowledge about data layout in files

Content-based data indexing

- Useful, but may require large amounts of meta-data

Scibox Approaches:

- Filter unnecessary data at producer-side via metadata (uploads)
- Merge overlapping subsets when multiple users share the same data (uploads)

Solutions for Minimizing Data Transfers for Data Sharing

Compression

- Helps, but compression ratio can be low for floating-point scientific data

Data selection

- Files: users can ask for subsets of data files, by specifying file offsets
- Requires knowledge about data layout in files

Content-based data indexing

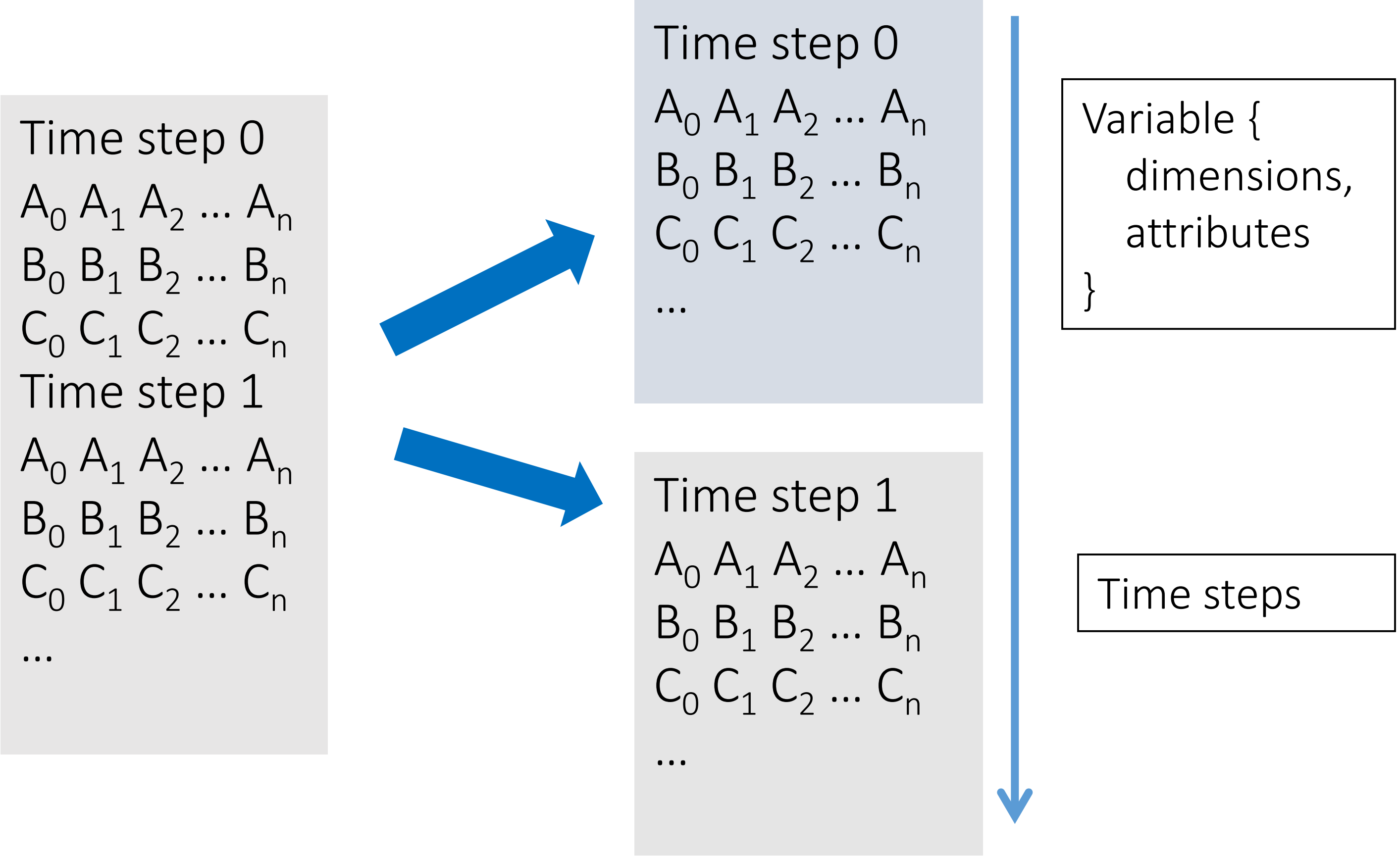
- Useful, but may require large amounts of meta-data

Scibox Approaches:

- Filter unnecessary data at producer-side via metadata (uploads)
- Merge overlapping subsets when multiple users share the same data (uploads)
- Minimize data sharing cost in cloud storage via new software protocol (downloads)

Challenge: How to Filter Data

Recall: scientific data is structured and meta-data rich



Time step 0
A_0 A_1 A_2 … A_n
B_0 B_1 B_2 … B_n
C_0 C_1 C_2 … C_n
Time step 1
A_0 A_1 A_2 … A_n
B_0 B_1 B_2 … B_n
C_0 C_1 C_2 … C_n
…

Time step 0
A_0 A_1 A_2 … A_n
B_0 B_1 B_2 … B_n
C_0 C_1 C_2 … C_n
…

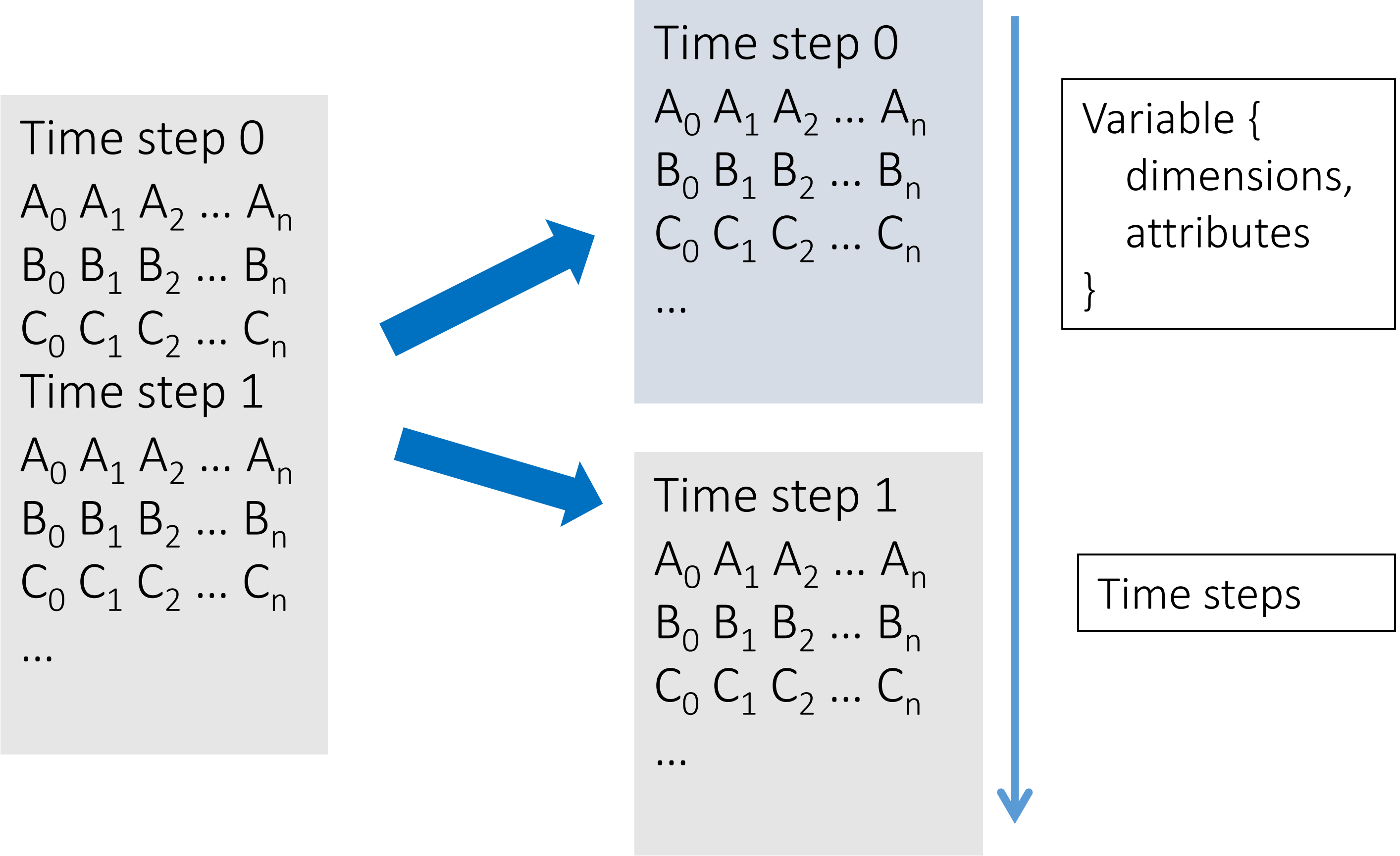
Time step 1
A_0 A_1 A_2 … A_n
B_0 B_1 B_2 … B_n
C_0 C_1 C_2 … C_n
…

```
Variable {
    dimensions,
    attributes
}
```

Time steps

Challenge: How to Filter Data

Recall: scientific data is structured and meta-data rich



Time step 0

A_0 A_1 A_2 … A_n

B_0 B_1 B_2 … B_n

C_0 C_1 C_2 … C_n

Time step 1

A_0 A_1 A_2 … A_n

B_0 B_1 B_2 … B_n

C_0 C_1 C_2 … C_n

…

Time step 0

A_0 A_1 A_2 … A_n

B_0 B_1 B_2 … B_n

C_0 C_1 C_2 … C_n

…

Time step 1

A_0 A_1 A_2 … A_n

B_0 B_1 B_2 … B_n

C_0 C_1 C_2 … C_n

…

```
Variable {
   dimensions,
   attributes
}
```

Time steps

Analytics users know what can/needs to be filtered

Outline

- Background and Motivation
- Problems and Challenges
- Design and Implementation
- Evaluation
- Conclusion and Future Work

Scibox Design

D(ata)R(eduction)-Function for data filtering

- Reduce cloud upload/download volumes
- Permit end users to identify the exact data needed for each specific analytics activity, using filters

Scibox Design

D(ata)R(eduction)-Function for data filtering

- Reduce cloud upload/download volumes
- Permit end users to identify the exact data needed for each specific analytics activity, using filters

Merge shared data objects before uploading

- Promote data sharing in the cloud to reduce data redundancy on the storage server and data volumes transferred across the network

Scibox Design

D(ata)R(eduction)-Function for data filtering

- Reduce cloud upload/download volumes
- Permit end users to identify the exact data needed for each specific analytics activity, using filters

Merge shared data objects before uploading

- Promote data sharing in the cloud to reduce data redundancy on the storage server and data volumes transferred across the network

Utilize ADIOS metadata-rich I/O methods

New ADIOS I/O transport

- Write output to cloud that can be directly read by subsequent, potentially remote data analytics or visualization codes
- Transparent on both the producer and consumer sides

Scibox Design

D(ata)R(eduction)-Function for data filtering

- Reduce cloud upload/download volumes
- Permit end users to identify the exact data needed for each specific analytics activity, using filters

Merge shared data objects before uploading

- Promote data sharing in the cloud to reduce data redundancy on the storage server and data volumes transferred across the network

Utilize ADIOS metadata-rich I/O methods

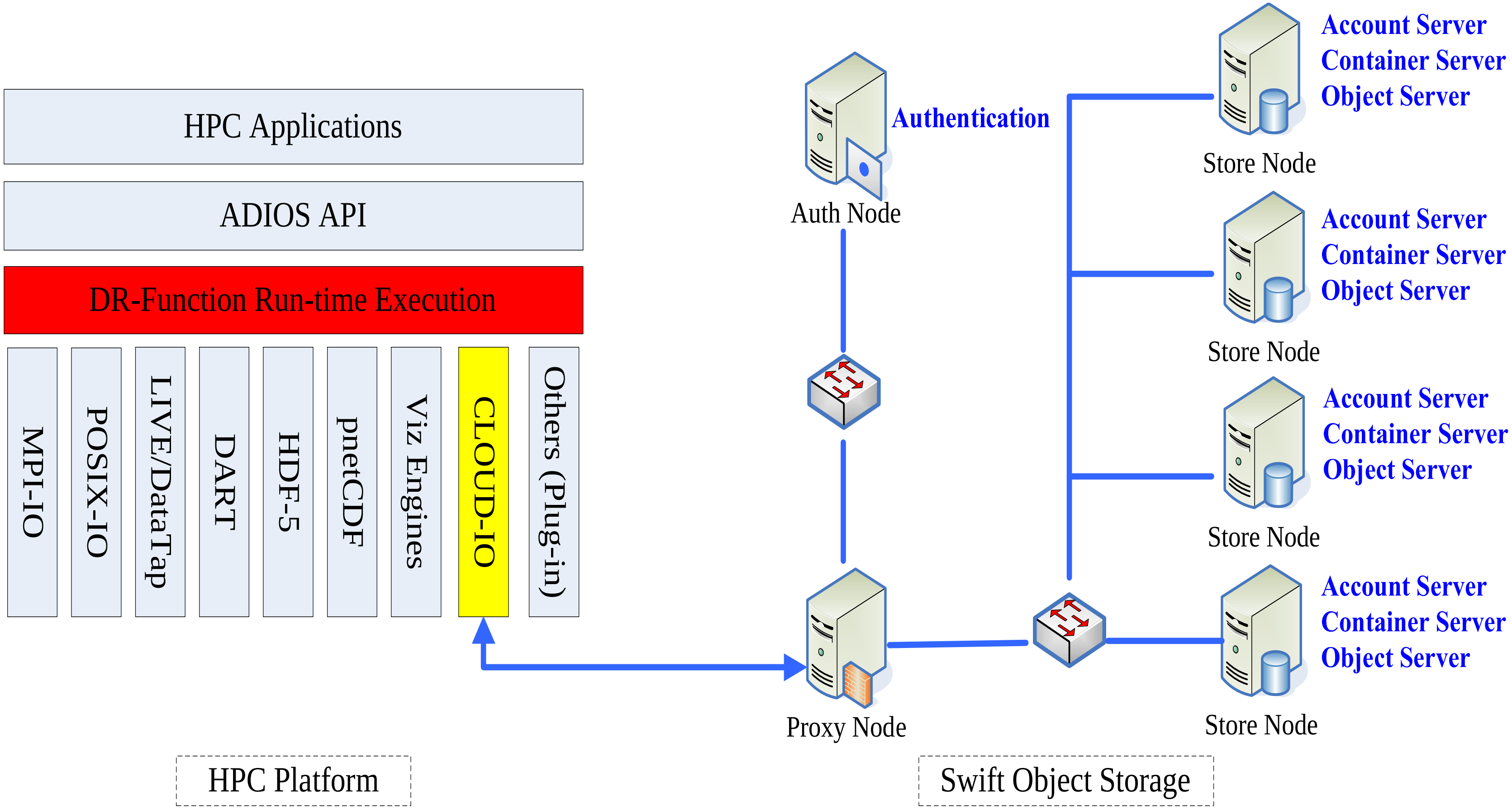
New ADIOS I/O transport

- Write output to cloud that can be directly read by subsequent, potentially remote data analytics or visualization codes
- Transparent on both the producer and consumer sides

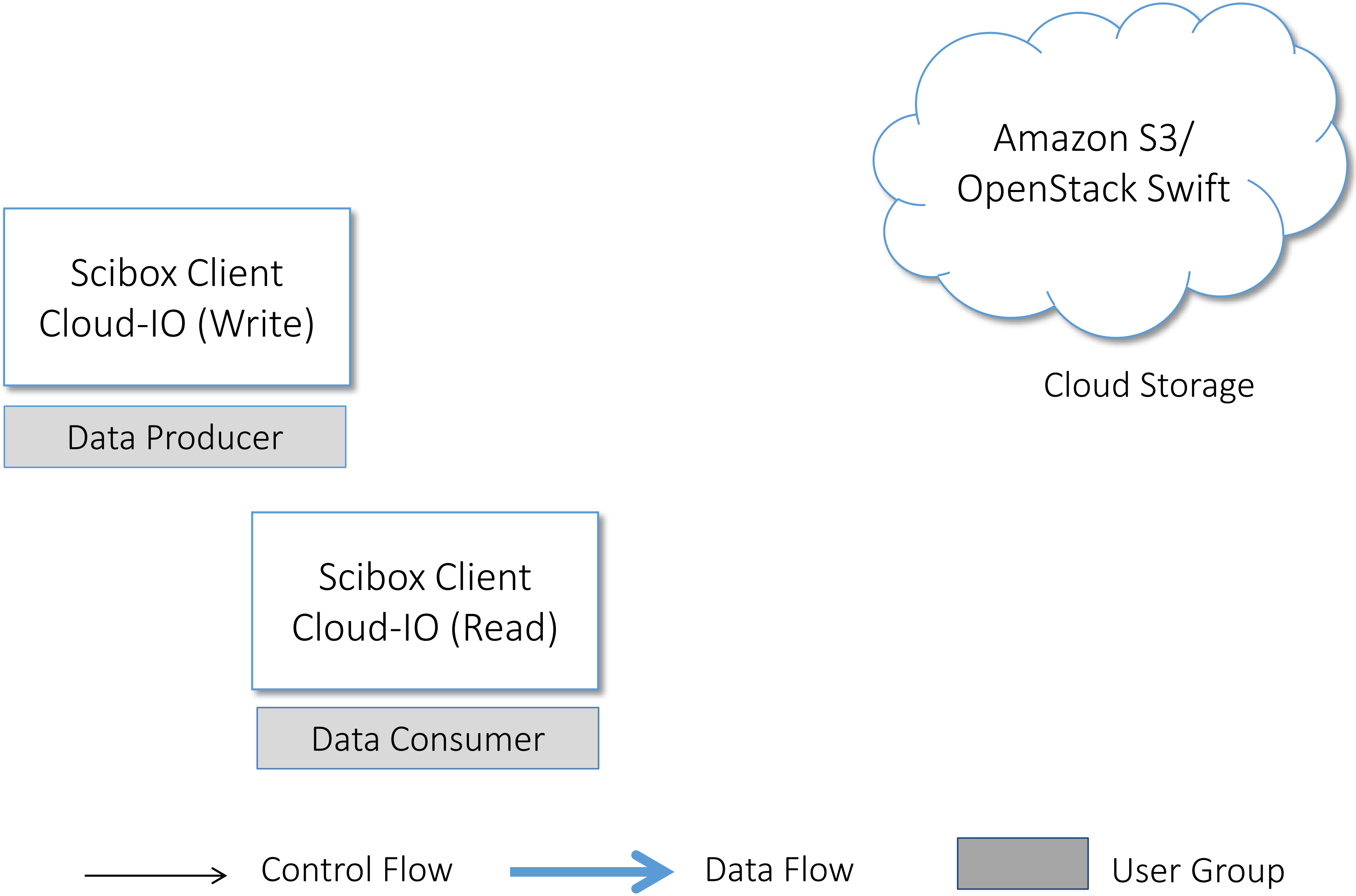
Partial object access for private cloud storage

- Patch the OpenStack Swift object store

Cloud-IO Transport



Scibox Architecture



Amazon S3/ OpenStack Swift

Cloud Storage

Scibox Client Cloud-IO (Write)

Data Producer

Scibox Client Cloud-IO (Read)

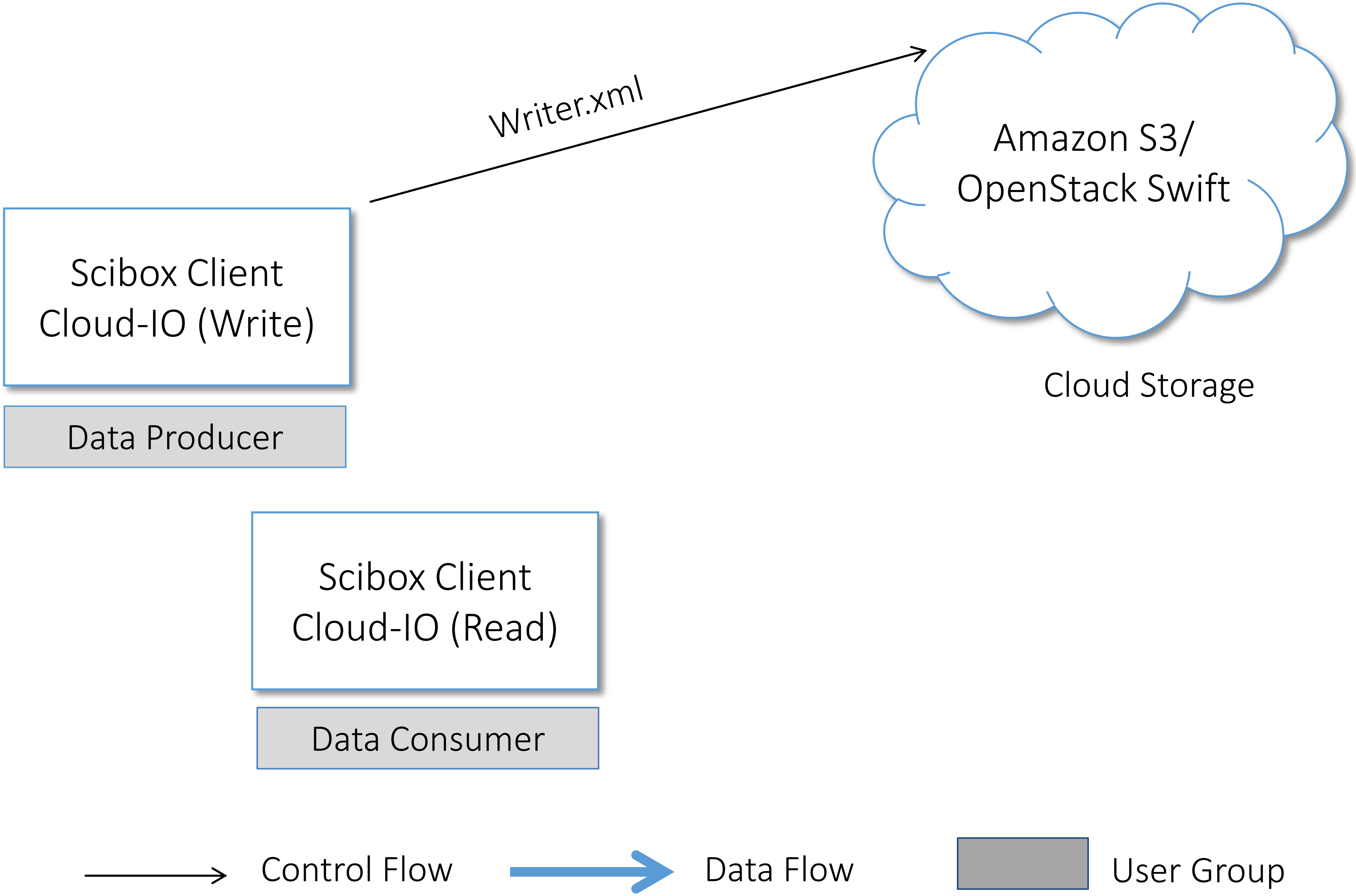
Data Consumer

Control Flow

Data Flow

User Group

Scibox Architecture



Writer.xml

Amazon S3/
OpenStack Swift

Cloud Storage

Scibox Client
Cloud-IO (Write)

Data Producer

Scibox Client
Cloud-IO (Read)

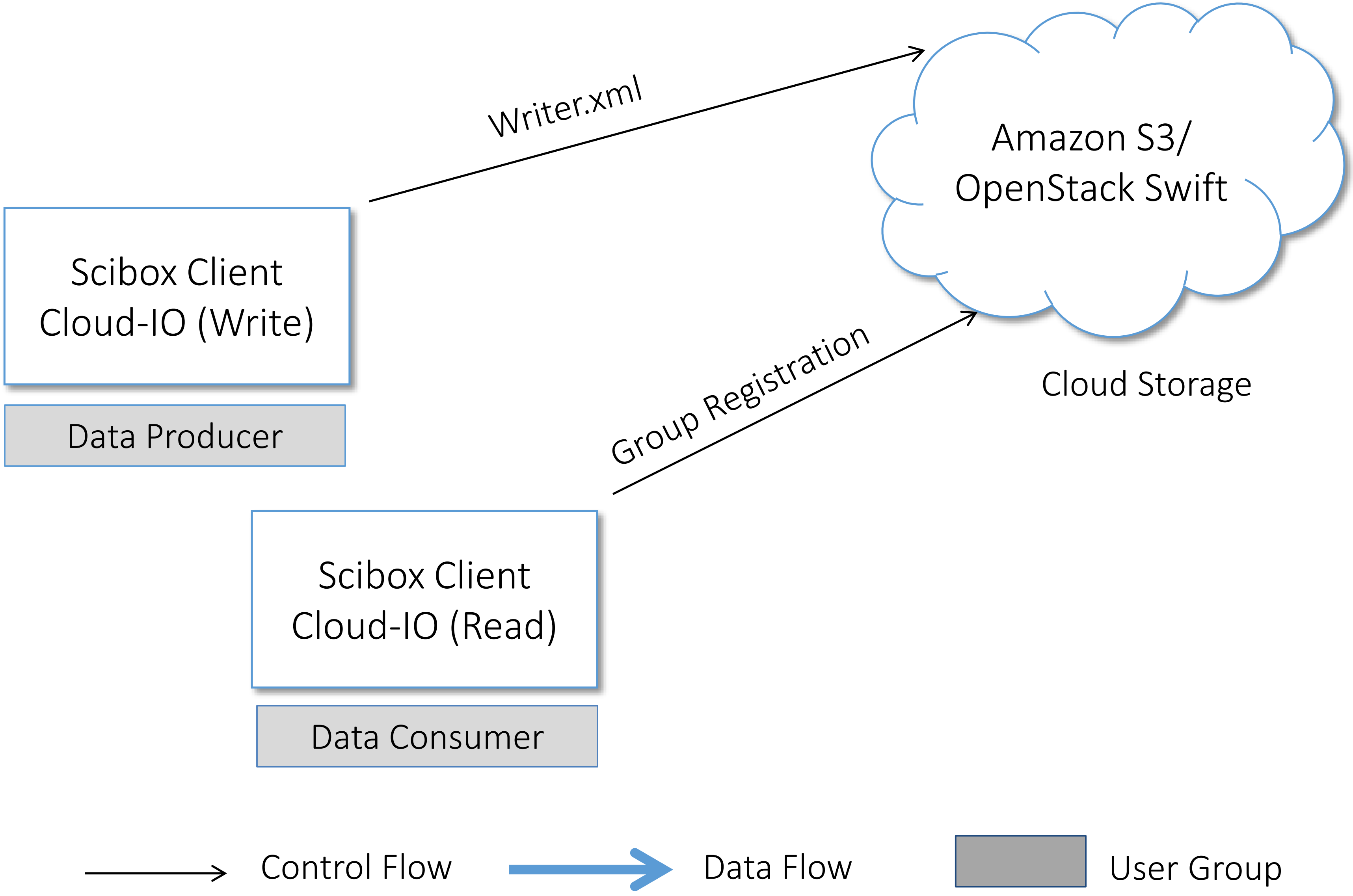
Data Consumer

Control Flow

Data Flow

User Group

Scibox Architecture


Writer.xml

Amazon S3/
OpenStack Swift

Cloud Storage

Scibox Client
Cloud-IO (Write)

Data Producer

Group Registration

Scibox Client
Cloud-IO (Read)

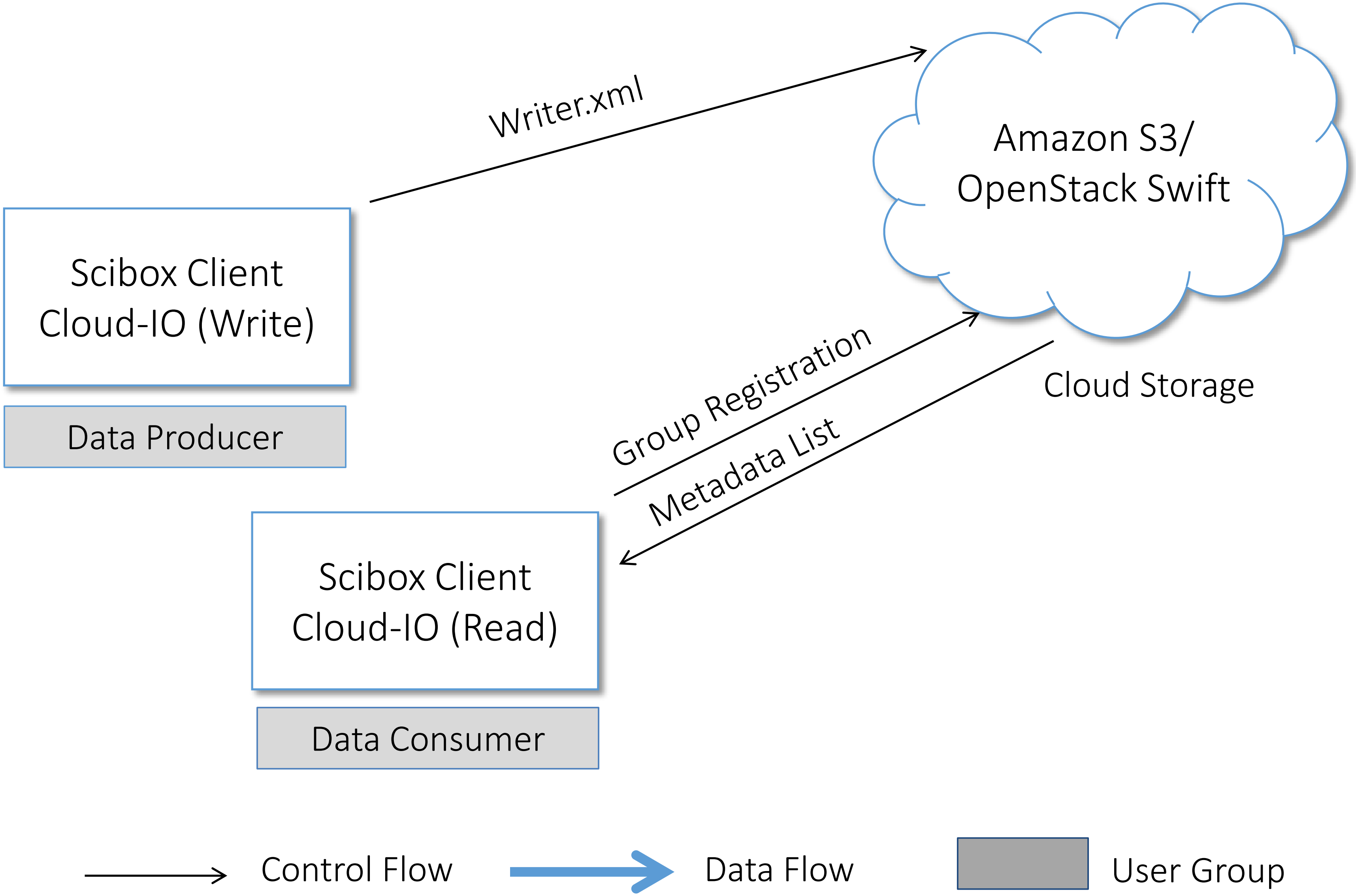
Data Consumer

Control Flow

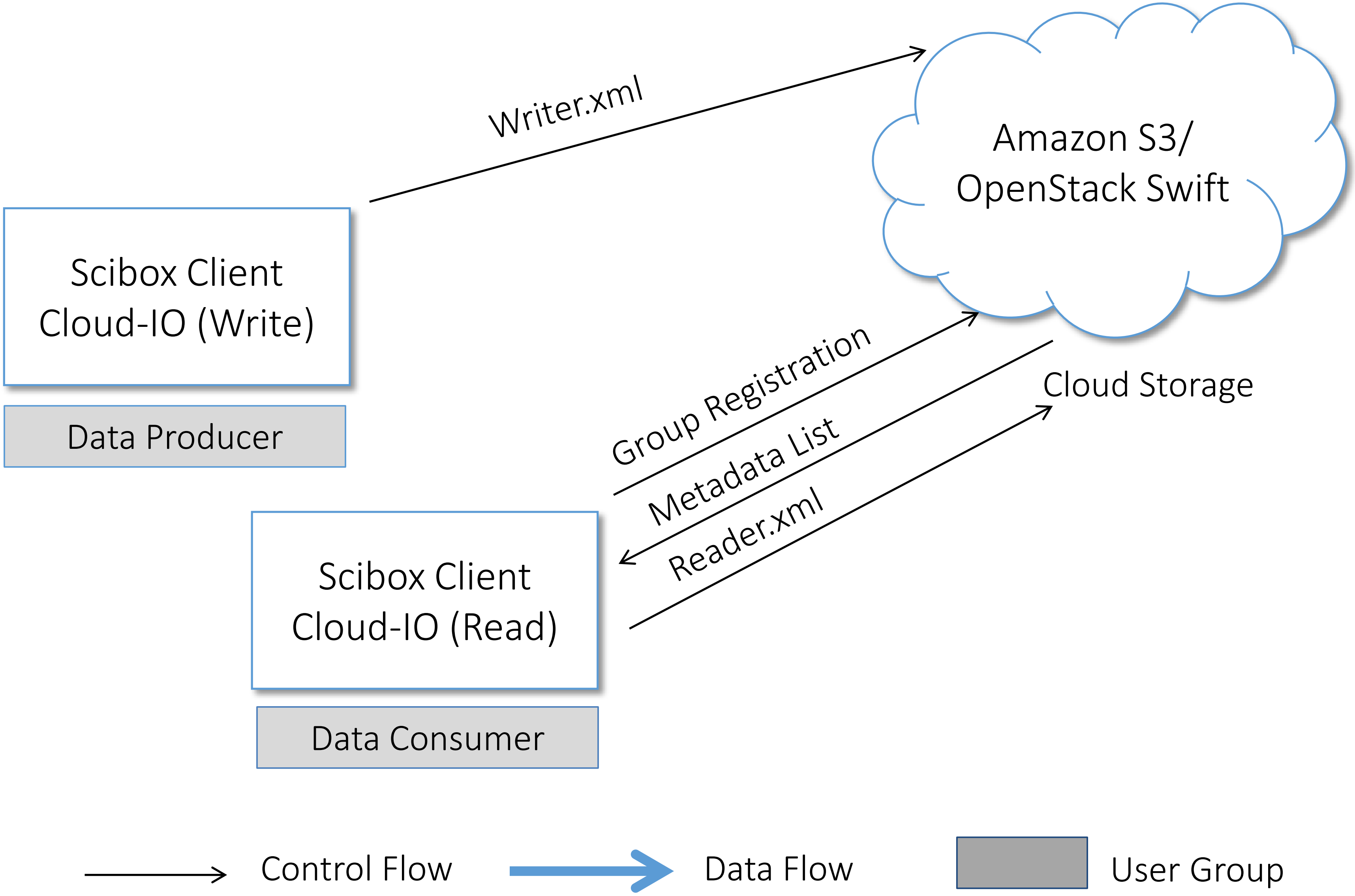
Data Flow

User Group

Scibox Architecture



Scibox Architecture



Writer.xml

Amazon S3/
OpenStack Swift

Cloud Storage

Scibox Client
Cloud-IO (Write)

Data Producer

Group Registration

Metadata List

Reader.xml

Scibox Client
Cloud-IO (Read)

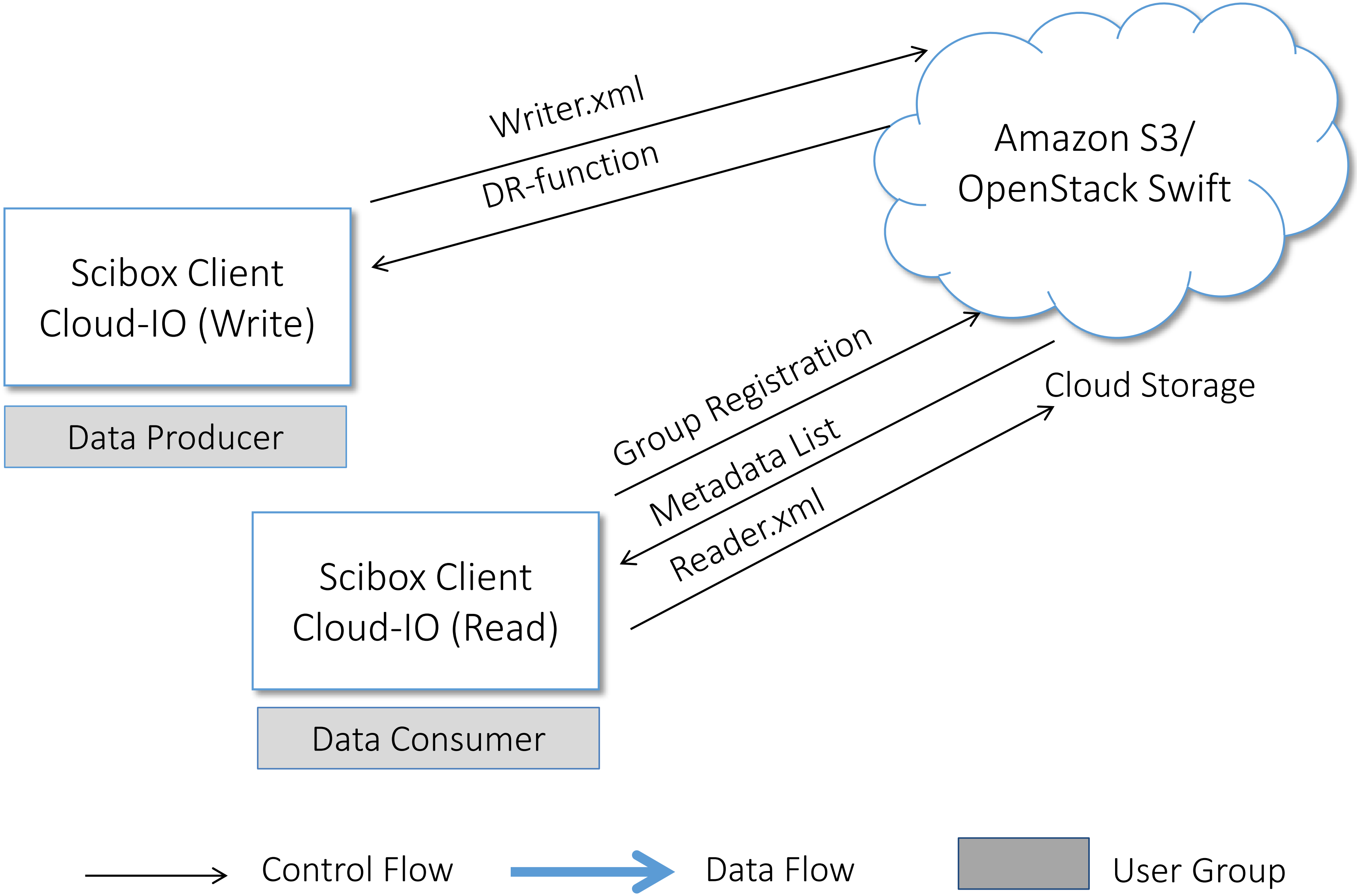
Data Consumer

Control Flow

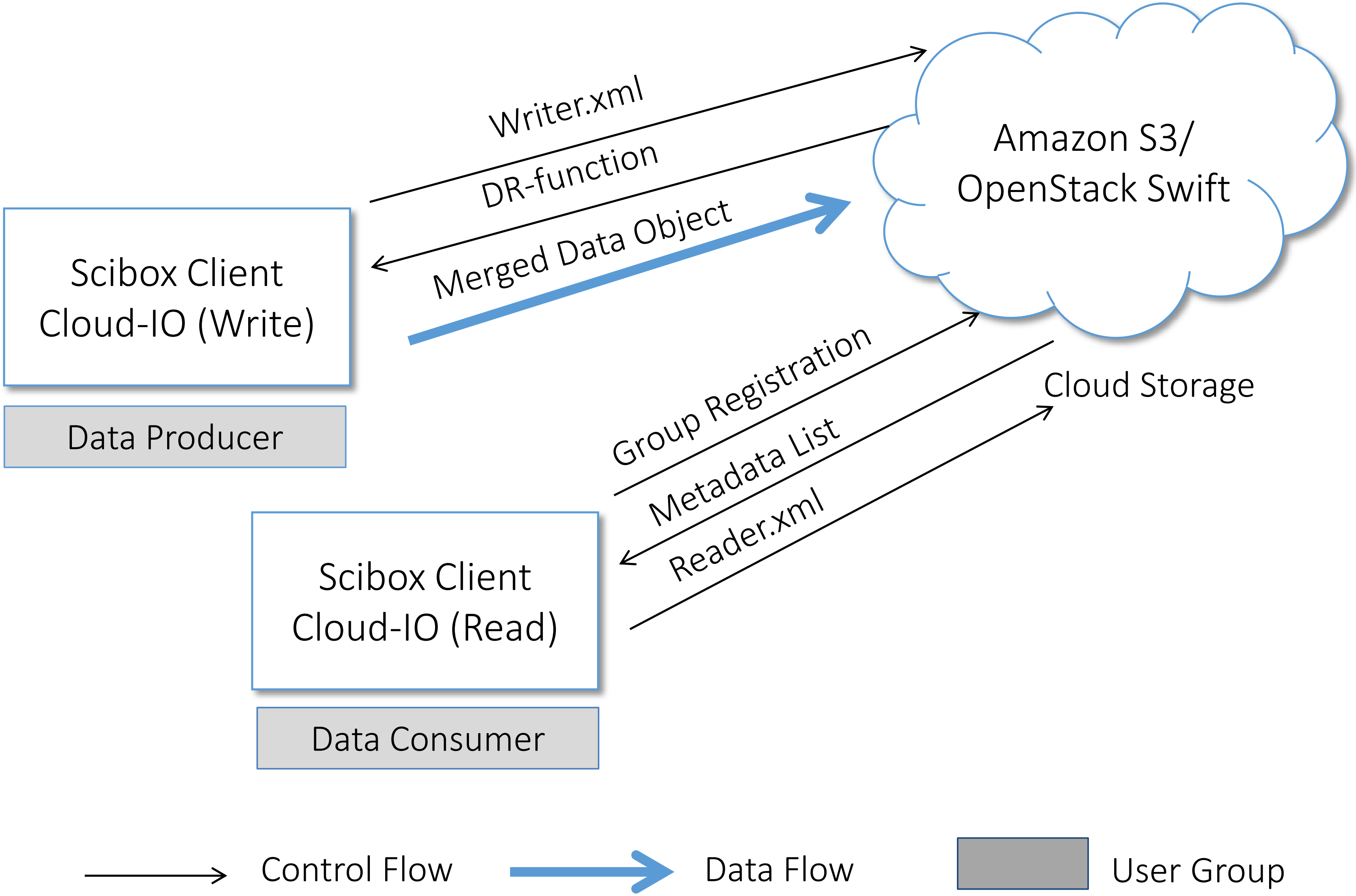
Data Flow

User Group

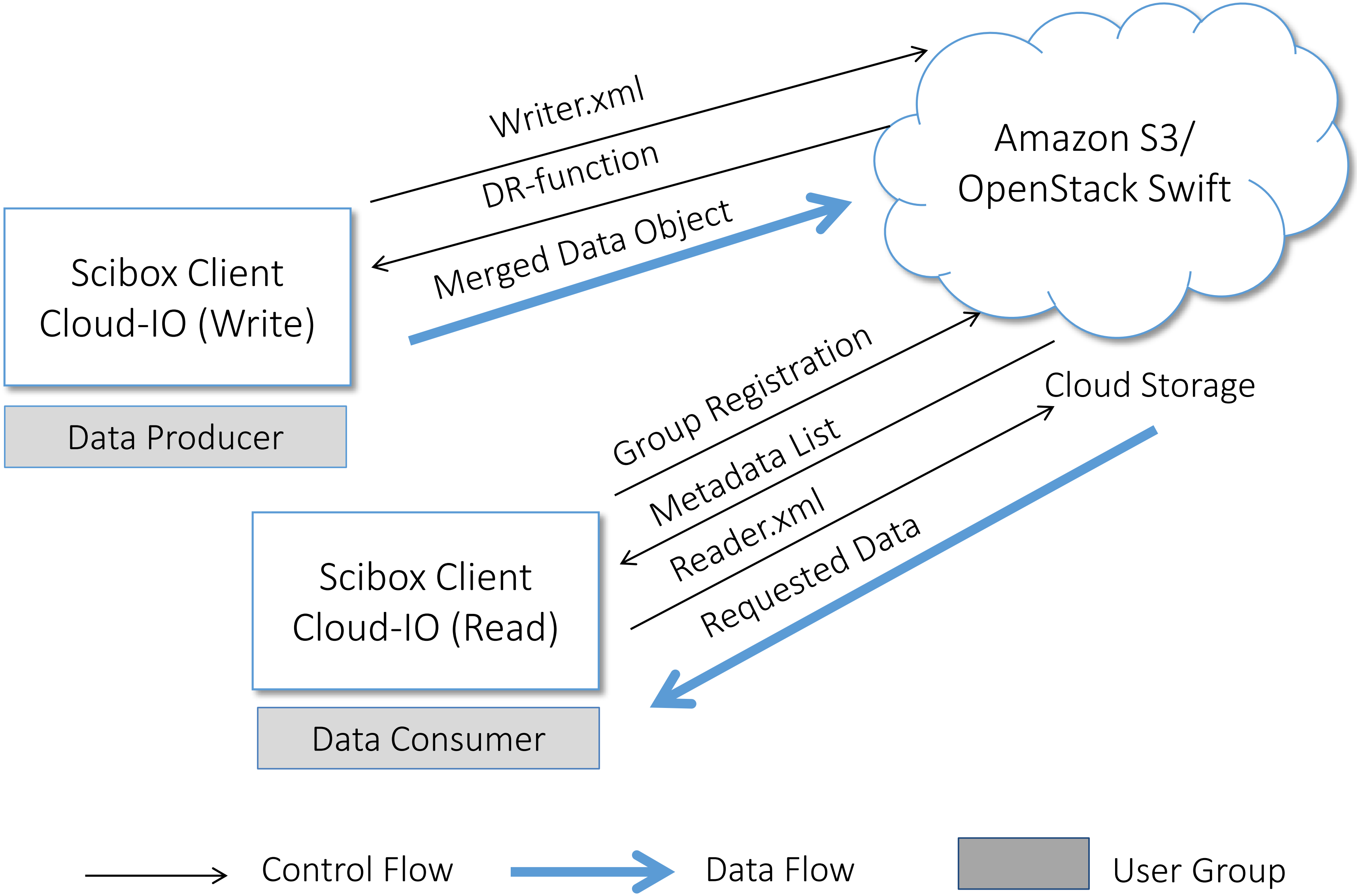
Scibox Architecture



Scibox Architecture



Scibox Architecture



Sample XML File

```
<?xml version="1.0"?>
<adios-config host-language="Fortran">

  <adios-group name="restart" coordination-communicator="comm">
    <var name="mype" type="integer"/>
    <var name="numberpe" type="integer"/>
    <var name="istep" type="integer"/>
    <var name="MIMAX_VAR" type="integer"/>
    <var name="NX" type="integer"/>
    <var name="NY" type="integer"/>
    <var name="zion0_1Darray"  gwrite="zion0"  type="double"
        dimensions="MIMAX_VAR”/>
    <var name="phi_2Darray"    gwrite="phi"  type="double" dimensions="NX, NY"/>
          <!– for reader.xml -->




  </adios-group>
<method group="restart"  method=”CLOUD-IO”>;</method>
<buffer size-MB="20" allocate-time="now"/>
</adios-config>
```

Sample XML File

```
<?xml version="1.0"?>
<adios-config host-language="Fortran">

   <adios-group name="restart" coordination-communicator="comm">
      <var name="mype" type="integer"/>
      <var name="numberpe" type="integer"/>
      <var name="istep" type="integer"/>
      <var name="MIMAX_VAR" type="integer"/>
      <var name="NX" type="integer"/>
      <var name="NY" type="integer"/>
      <var name="zion0_1Darray"  gwrite="zion0"  type="double"
            dimensions="MIMAX_VAR”/>
      <var name="phi_2Darray"   gwrite="phi"  type="double" dimensions="NX, NY"/>
               <!– for reader.xml -->
      <rd type=8 name="phi_2Darray”
                    cod=“int i; double sum = 0.0;
                    for(i = 0; i<input.count; i= i+1)
                    { sum = sum + input.vals[i]; }
                     return sum;” />
   </adios-group>
<method group="restart"  method=”CLOUD-IO”>;</method>
<buffer size-MB="20" allocate-time="now"/>
</adios-config>
```

Data Reduction (DR) Functions

Definition

- Function defined by end-users that can transform/reduce data to prepare it for cloud storage

Data Reduction (DR) Functions

Definition

- Function defined by end-users that can transform/reduce data to prepare it for cloud storage

Creation

- Programmed by end users
- Generated from higher level descriptions
- Derived from users’ I/O access patterns

Data Reduction (DR) Functions

Definition

- Function defined by end-users that can transform/reduce data to prepare it for cloud storage

Creation

- Programmed by end users
- Generated from higher level descriptions
- Derived from users' I/O access patterns

Current implementation

- Customized CoD (C on Demand)

  require producer-side computational resources

- DR-function library

  same DR-function specified by multiple clients, will be executed only once, and its output data will be reused for multiple consumers

DR-functions Provided by SciBox

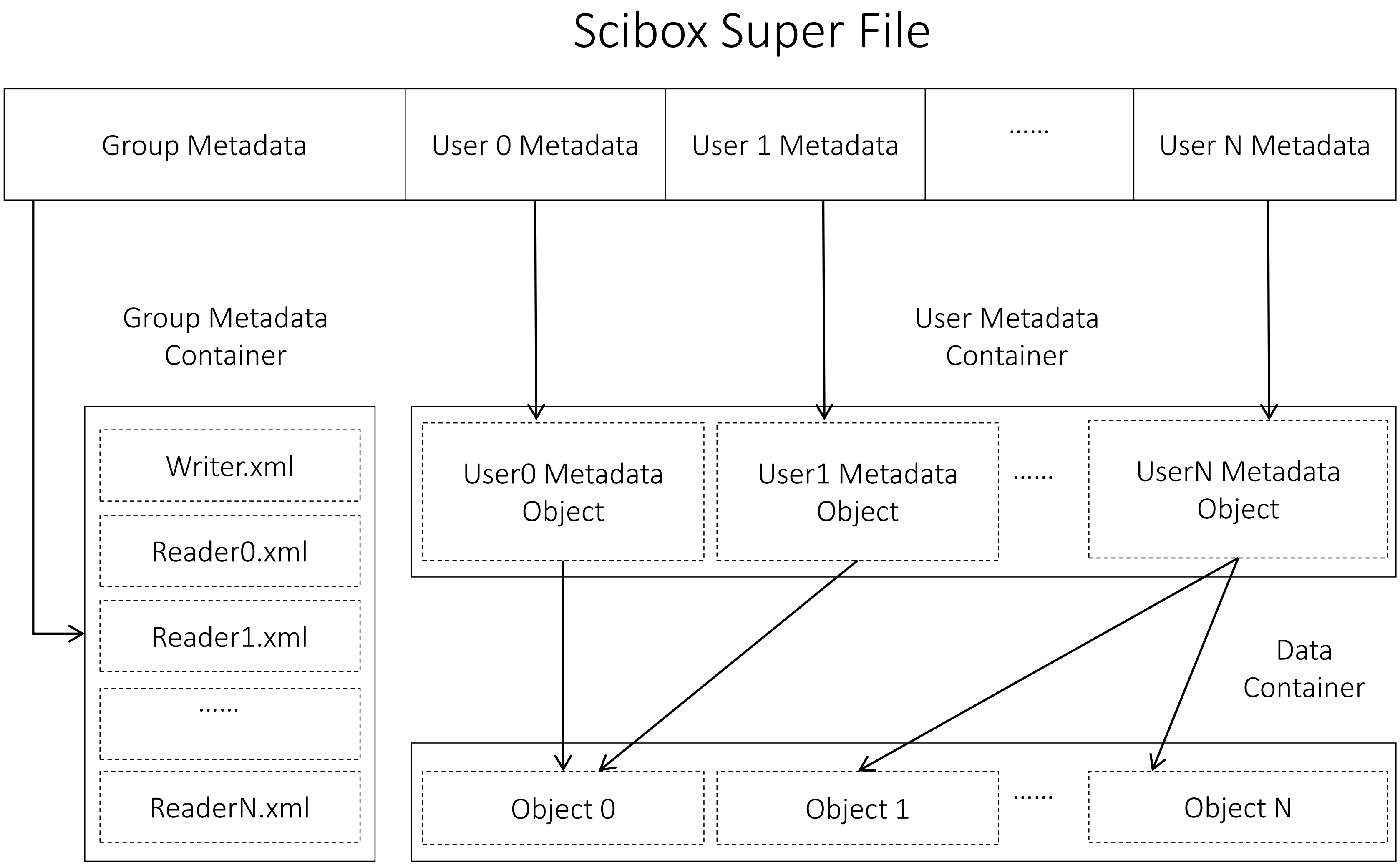
Type	Description	Example
DR1	Max(variable)	Max(var_double_2Darray)
DR2	Min(variable)	Min(var_double_2Darray)
DR3	Mean(variable)	Mean(var_double_2Darray)
DR4	Range(variable, dimension, start_pos, end_pos)	Range(var_int_1Darray, 1, 100, 1000)
DR5	Select(variable, threshold1, threshold2)	Select var.value where var.value in (threshold1, threshold2)
DR6	Select(variable, DR_Function1, DR_Function2)	Select var.value where var.value ≥ Mean(var)
DR7	Select(variable1, variable2, threshold1, threshold2)	Select var2.value where var1.value in (threshold1, threshold2)
DR8	Self defined function	Double proc(cod_exec_context ec, input_type * input, int k, int m) {int l; intj; double sum = 0.0; double average=0.0; for(i=0;i<m;i++)sum+=input.tmpbuf[i+k*m];average=sum/m; resturn average;}

DR-functions Provided by SciBox

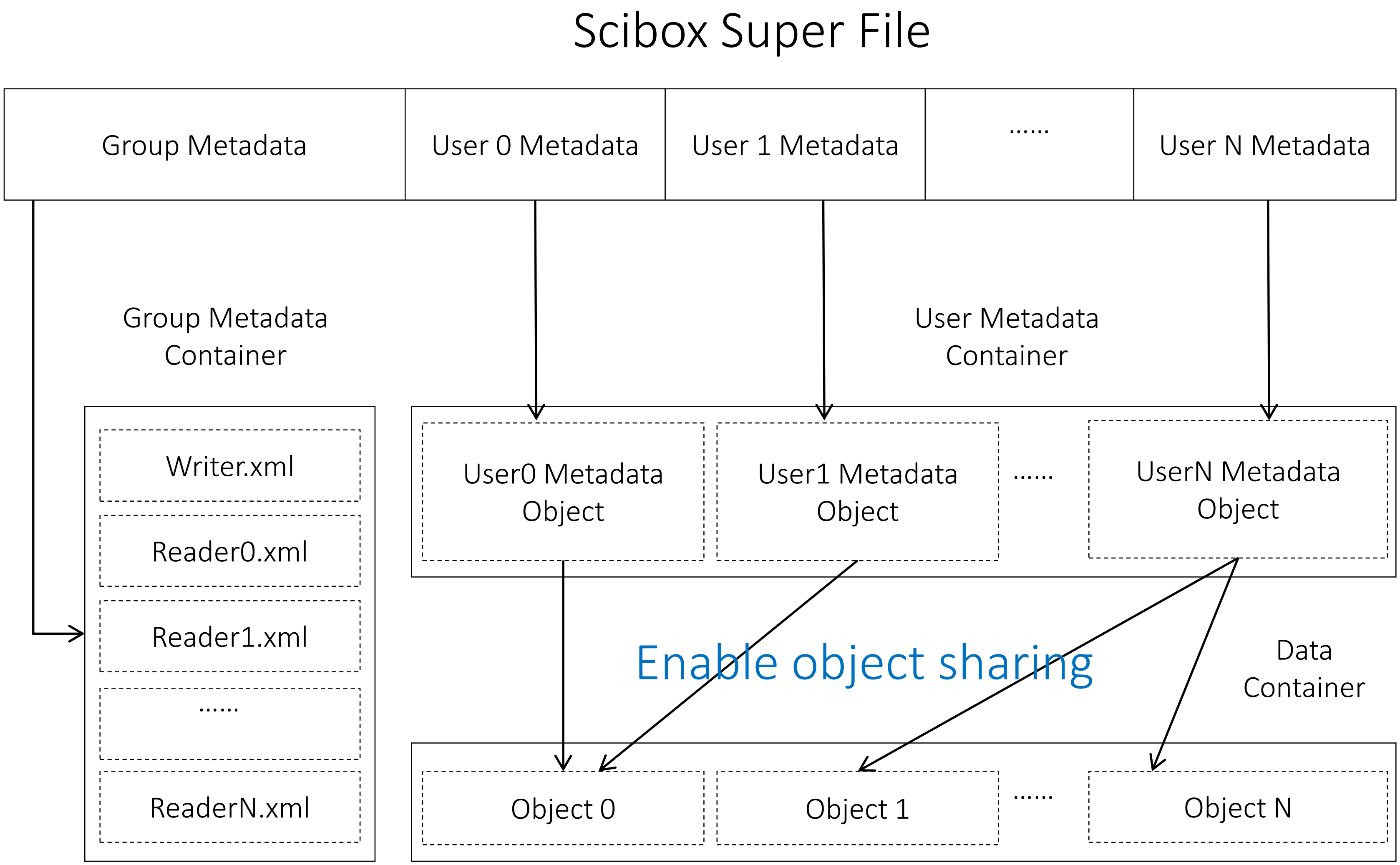
Type	Description	Example
DR1	Max(variable)	Max(var_double_2Darray)
DR2	Min(variable)	Min(var_double_2Darray)
DR3	Mean(variable)	Mean(var_double_2Darray)
DR4	Range(variable, dimension, start_pos, end_pos)	Range(var_int_1Darray, 1, 100, 1000)
DR5	Select(variable, threshold1, threshold2)	Select var.value where var.value in (threshold1, threshold2)
DR6	Select(variable, DR_Function1, DR_Function2)	Select var.value where var.value ≥ Mean(var)
DR7	Select(variable1, variable2, threshold1, threshold2)	Select var2.value where var1.value in (threshold1, threshold2)
DR8	Self defined function	Double proc(cod_exec_context ec, input_type * input, int k, int m) {int l; intj; double sum = 0.0; double average=0.0; for(i=0;i<m;i++)sum+=input.tmpbuf[i+k*m];average=sum/m; resturn average;}

C on Demand (CoD):
Consumer: a string
Producer:
1. registration
2. compile and execute on demand.

Data Object Management



Data Object Management



Determination of Object Size

Merge overlapping data sets

- reduce upload data size

Partial object access

- Current Amazon S3 and OpenStack Swift stores do not support this
- Users have to download the whole object, even if only a small portion of its data is needed

Determination of Object Size

Merge overlapping data sets

- reduce upload data size

Partial object access

- Current Amazon S3 and OpenStack Swift stores do not support this
- Users have to download the whole object, even if only a small portion of its data is needed

Two approaches used in Scibox

- Private cloud

  modify the software to enable partial object access

  Object size is determined by predicting upload throughput

Determination of Object Size

Merge overlapping data sets

- reduce upload data size

Partial object access

- Current Amazon S3 and OpenStack Swift stores do not support this
- Users have to download the whole object, even if only a small portion of its data is needed

Two approaches used in Scibox

- Private cloud

  modify the software to enable partial object access

  Object size is determined by predicting upload throughput

- Public cloud

  limit object size considering storage pricing

  Object size is determined by comparing the cost w/ sharing and w/o sharing

Cost Model

Definition

α: $/GB of standard cloud storage

β: $/GB of data transfer into cloud

γ: $/GB of data transfer out from cloud

Assumption

n clients request Data_1, Data_2, … Data_n respectively

Cost Model

Definition

α: $/GB of standard cloud storage

β: $/GB of data transfer into cloud

γ: $/GB of data transfer out from cloud

Assumption

n clients request Data_1, Data_2, … Data_n respectively

Without sharing:

$$Cost_{nosharing} = (\alpha + \beta + \gamma) * \sum_{i=1}^{n} Data_i$$

Cost Model

Definition

α: $/GB of standard cloud storage

β: $/GB of data transfer into cloud

γ: $/GB of data transfer out from cloud

Assumption

n clients request Data_1, Data_2, … Data_n respectively

Without sharing:

$$Cost_{nosharing} = (\alpha + \beta + \gamma) * \sum_{i=1}^{n} Data_i$$

With sharing (n objects can be merged into one):

$$Cost_{sharing} = (\alpha + \ \beta) * Size + r * \sum_{i=1}^{n} Size$$

Cost Model

Definition

α: \$/GB of standard cloud storage

β: \$/GB of data transfer into cloud

γ: \$/GB of data transfer out from cloud

Assumption

n clients request Data_1, Data_2, … Data_n respectively

Without sharing:

$$Cost_{nosharing} = (\alpha + \beta + \gamma) * \sum_{i=1}^{n} Data_i$$

With sharing (n objects can be merged into one):

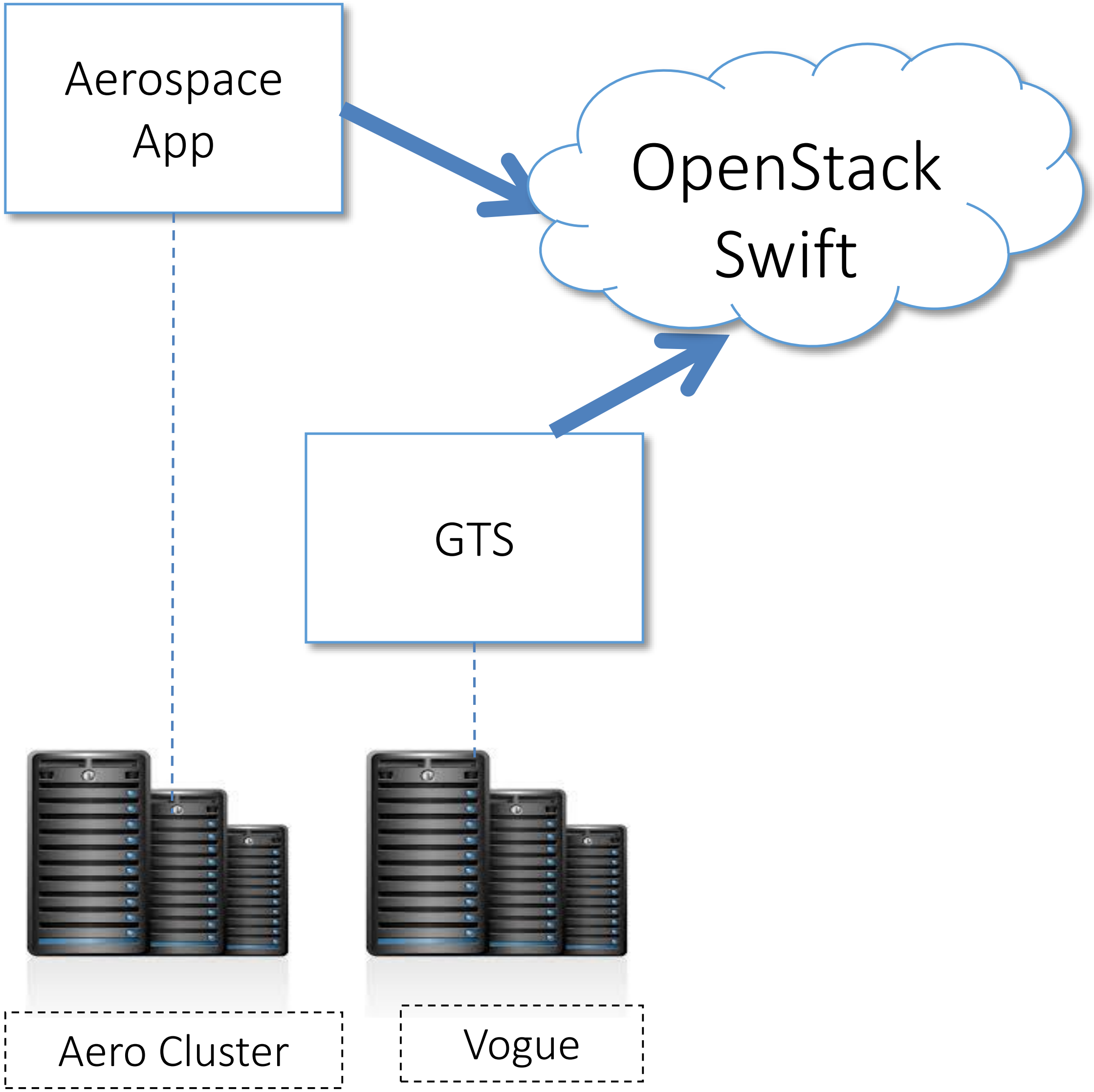
$$Cost_{sharing} = (\alpha + \ \beta) * Size + r * \sum_{i=1}^{n} Size$$

$$Size \leq \frac{(\alpha + \beta + \gamma) * \sum_{i=1}^{n} Data_i}{\alpha + \beta + n\gamma}$$

Outline

- Background and Motivation
- Problems and Challenges
- Design and Implementation
- Evaluation
- Conclusion and Future Work

Experimental Setup



Aerospace App

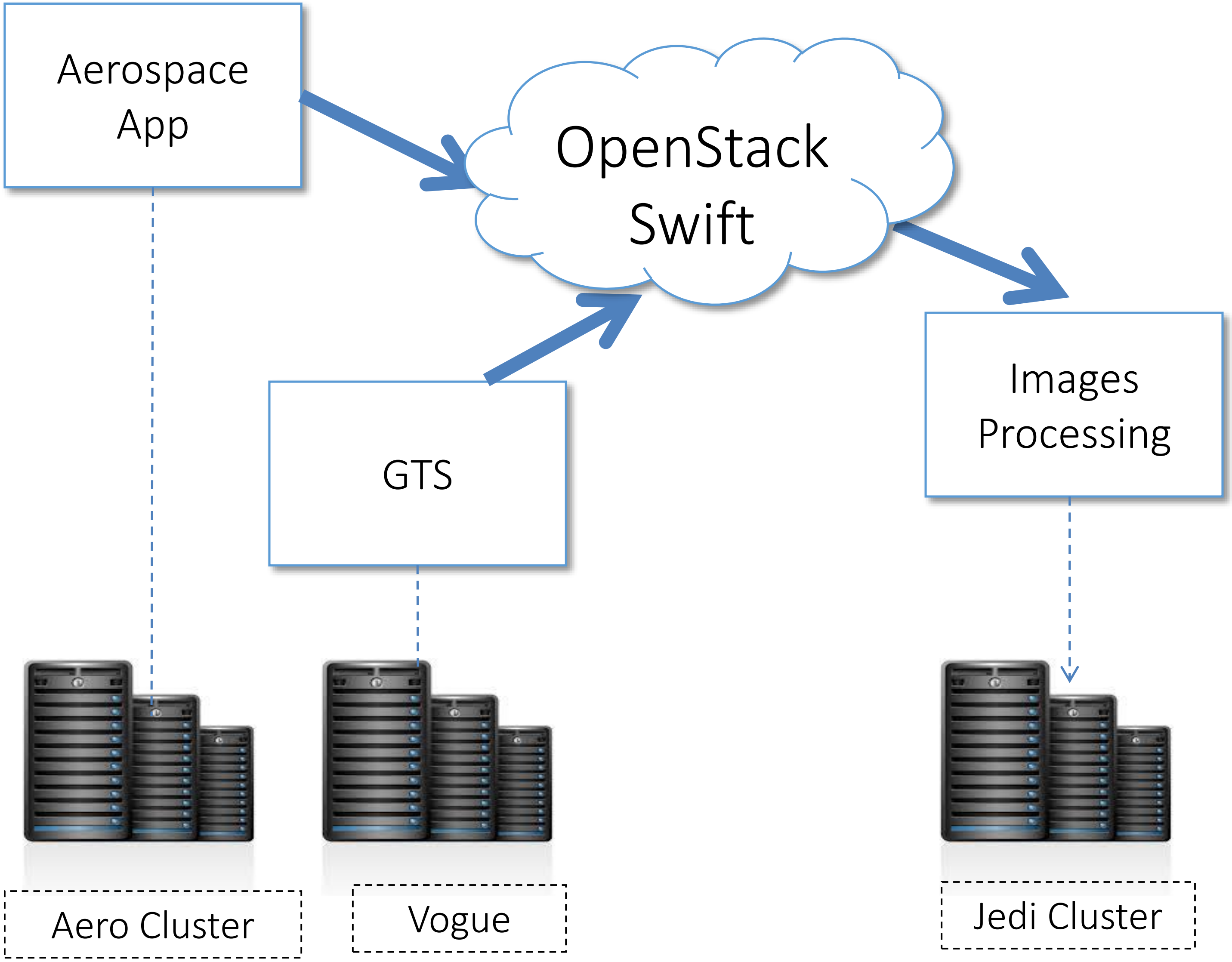
OpenStack Swift

GTS

Aero Cluster

Vogue

Experimental Setup



Aerospace App

OpenStack Swift

GTS

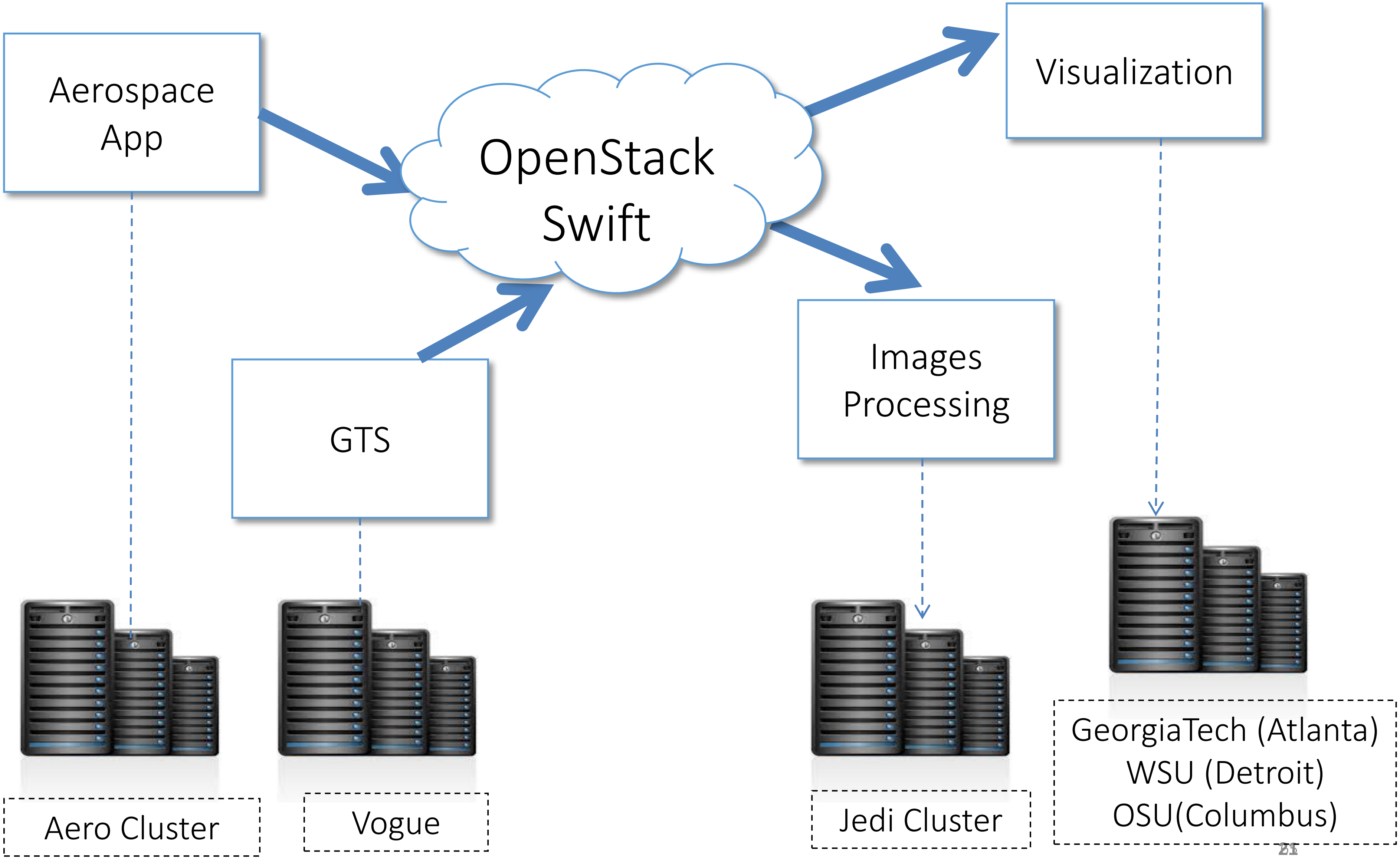
Images Processing

Aero Cluster

Vogue

Jedi Cluster

Experimental Setup



Aerospace App

GTS

OpenStack Swift

Visualization

Images Processing

Aero Cluster

Vogue

Jedi Cluster

GeorgiaTech (Atlanta)
WSU (Detroit)
OSU(Columbus)

Workloads

Synthetic Workloads

- 10 variables shared by multiple consumers
- 1,000 requests generated by each consumer
- 8 types of DR functions, uniformly distributed
- 1 data producer serving 1,000 requests x #client servers
- 3 self-defined DR-functions:

  FFT, histogram diagnosis, and average of row values of a matrix

Workloads

Synthetic Workloads

- 10 variables shared by multiple consumers
- 1,000 requests generated by each consumer
- 8 types of DR functions, uniformly distributed
- 1 data producer serving 1,000 requests x #client servers
- 3 self-defined DR-functions:

  FFT, histogram diagnosis, and average of row values of a matrix

Real Workloads

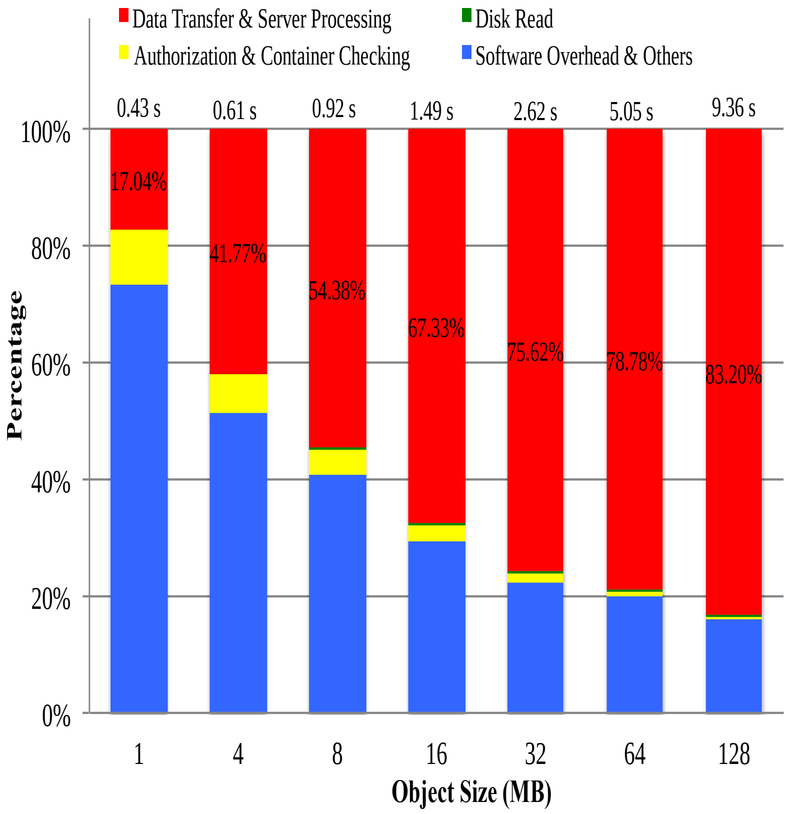
- GTS workload

  128 parallel processes, consumers are from 3 different states in USA

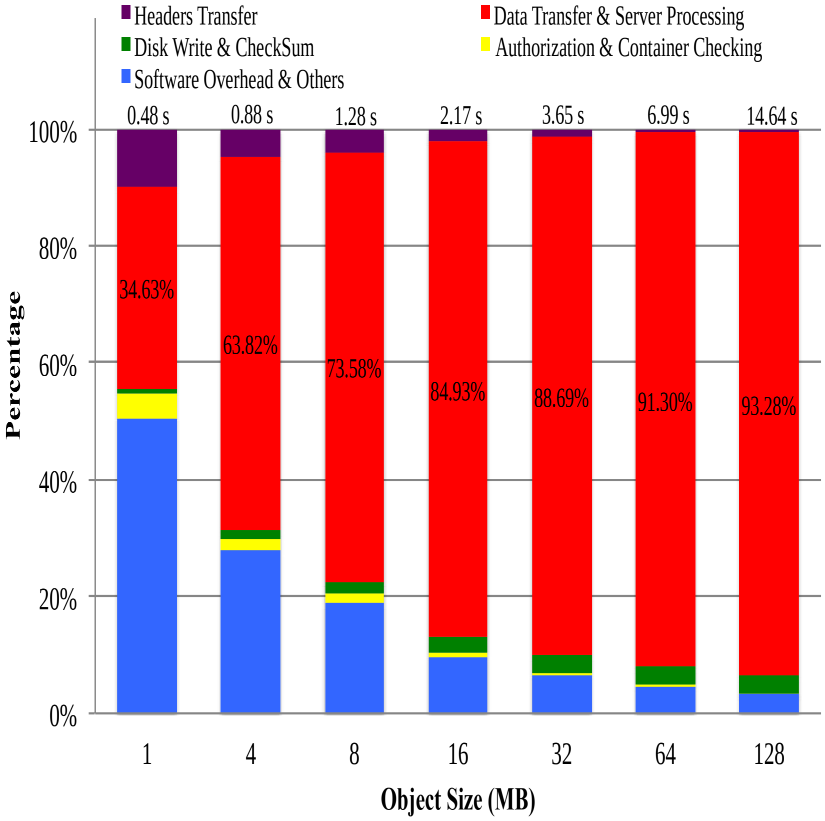
- Combustion workload

  10, 000 512X512 12-bit double framed images (~1.5 MB per image)

Latency Breakdown of Swift Object Store

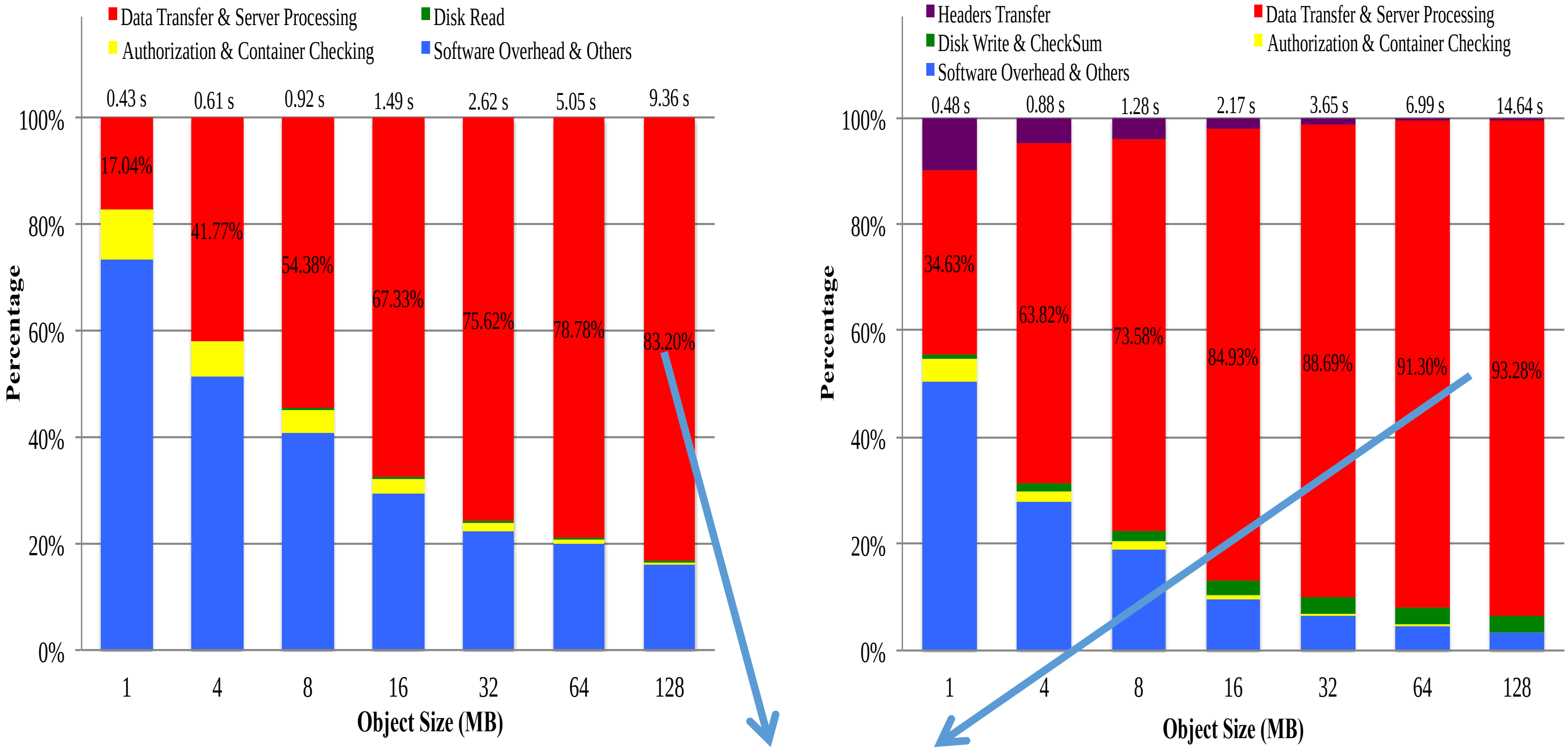


Put



Get

Latency Breakdown of Swift Object Store

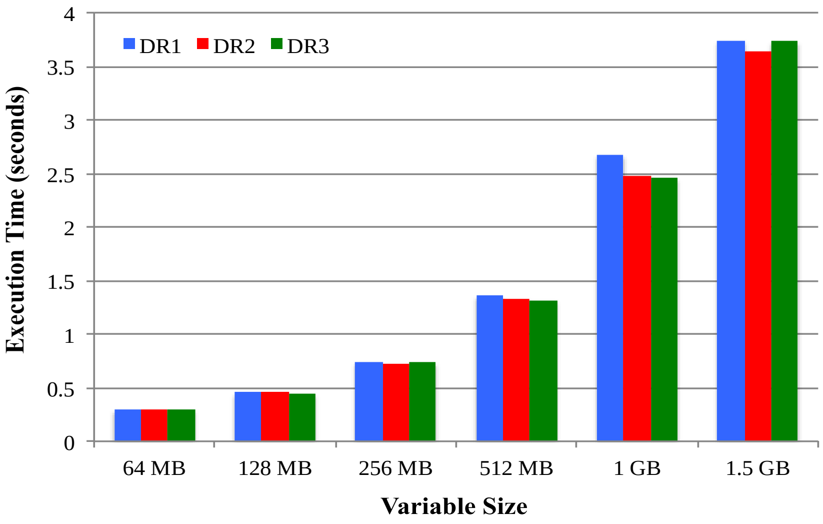


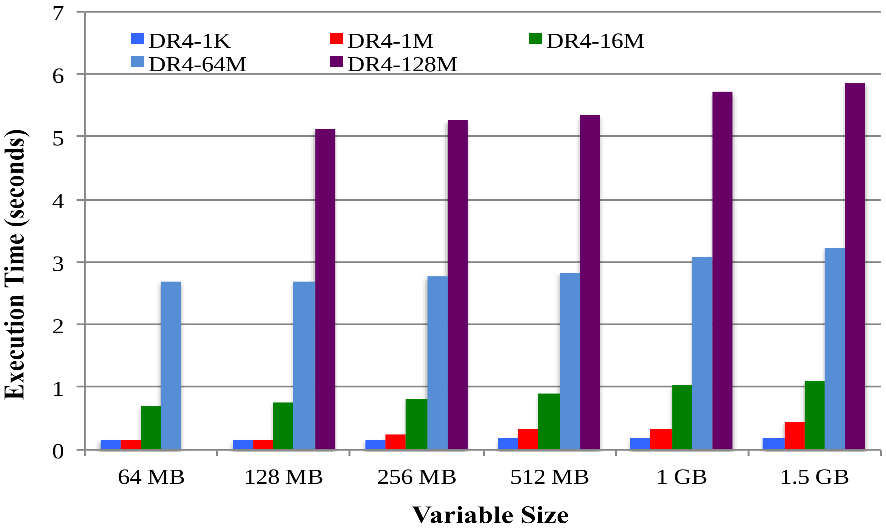
Put

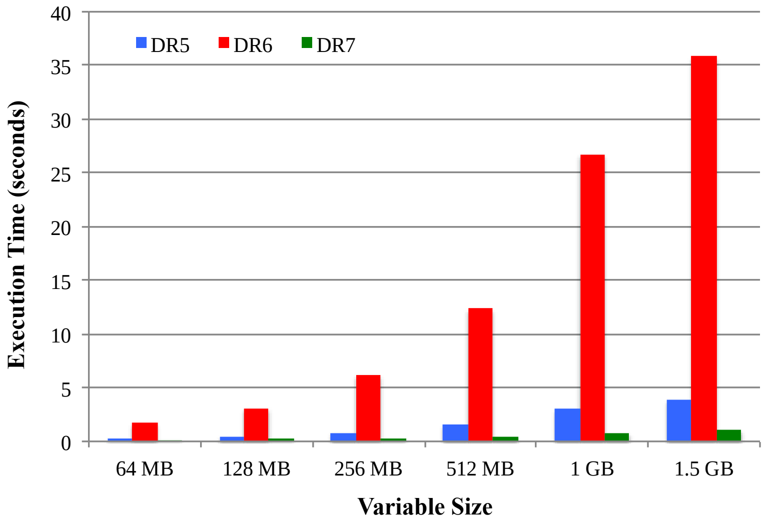
Data transfer dominates the request latency.

Get

Execution Time of DR-functions







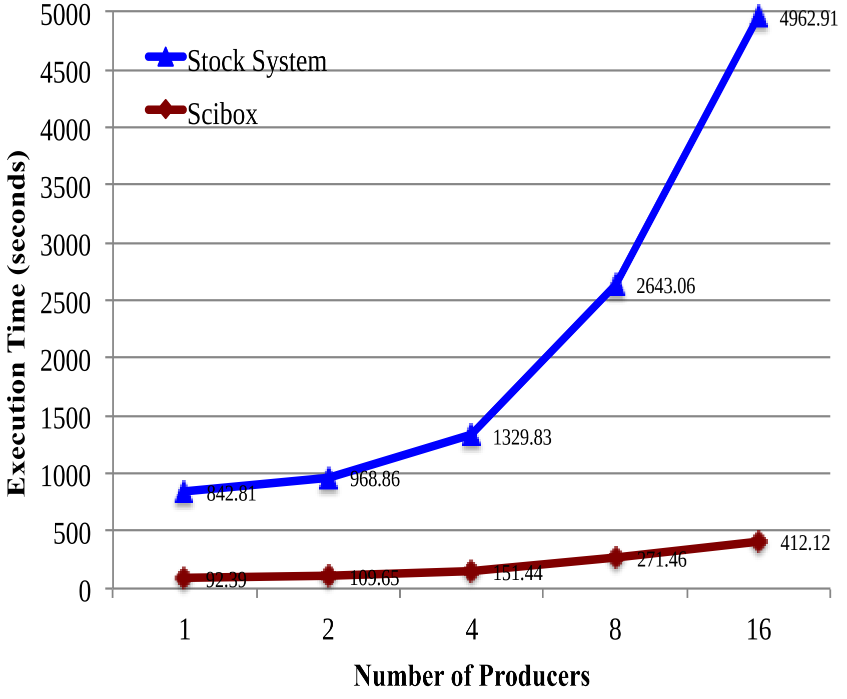
Recall: DR6:var.value where var.value>Mean(var).

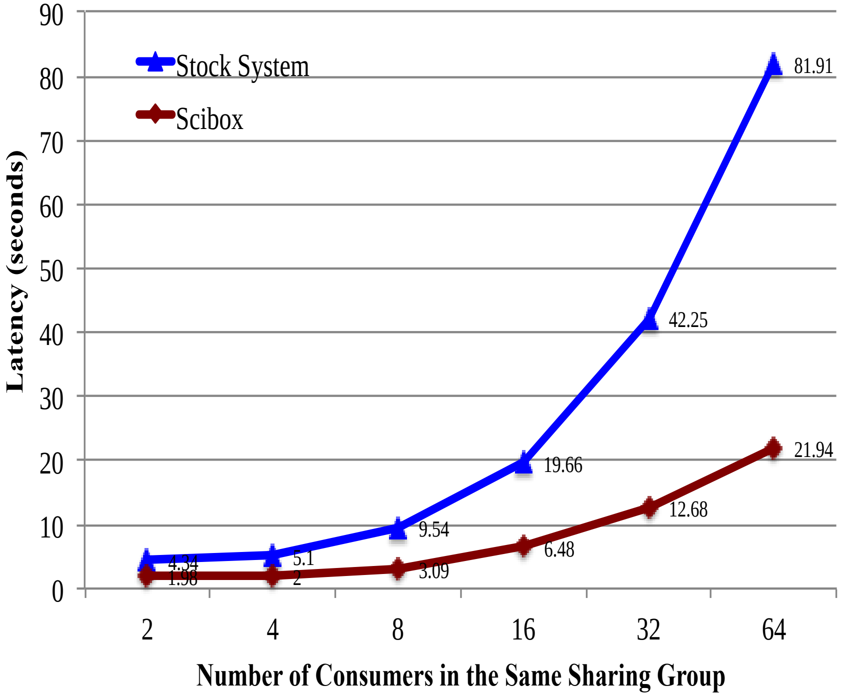
Double scan of input data.

Recall: DR7:Select var2.value where var1.value in (r1, r2).

var1 is small.

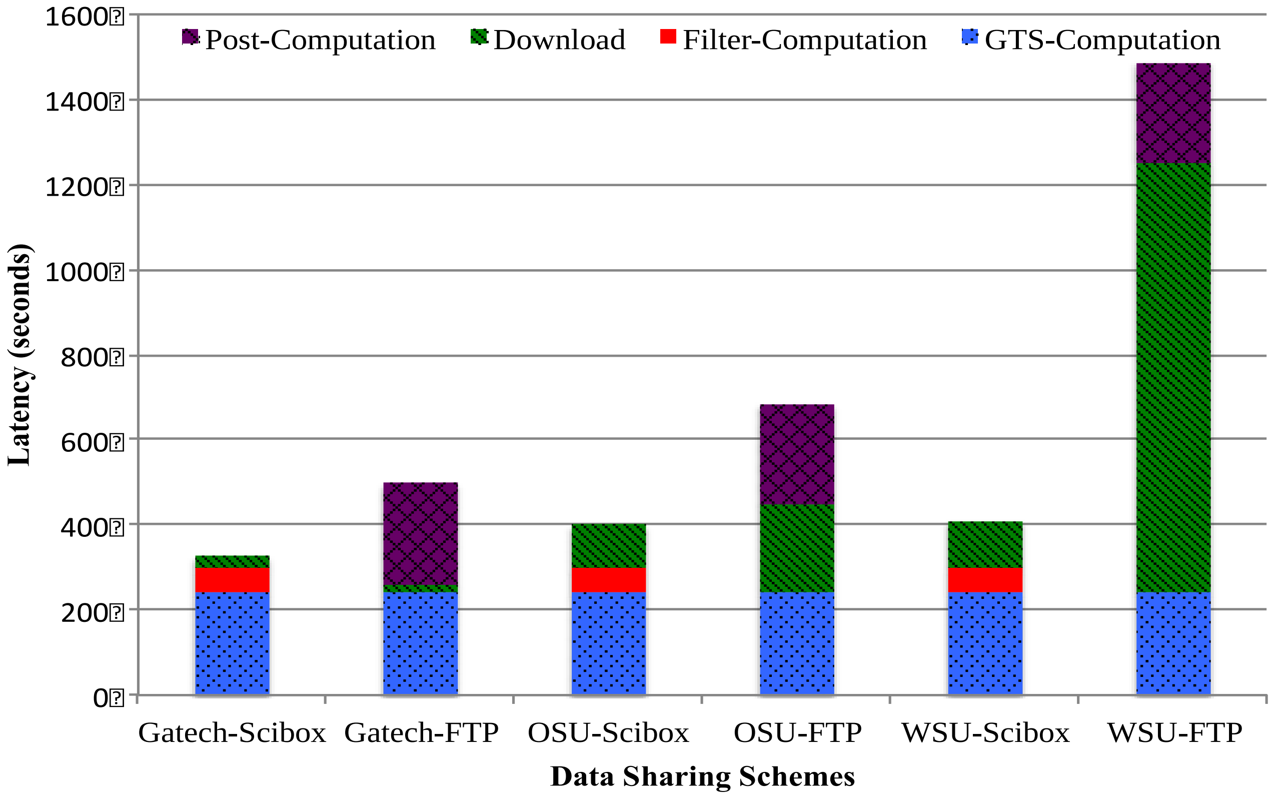
SciBox System Scalability





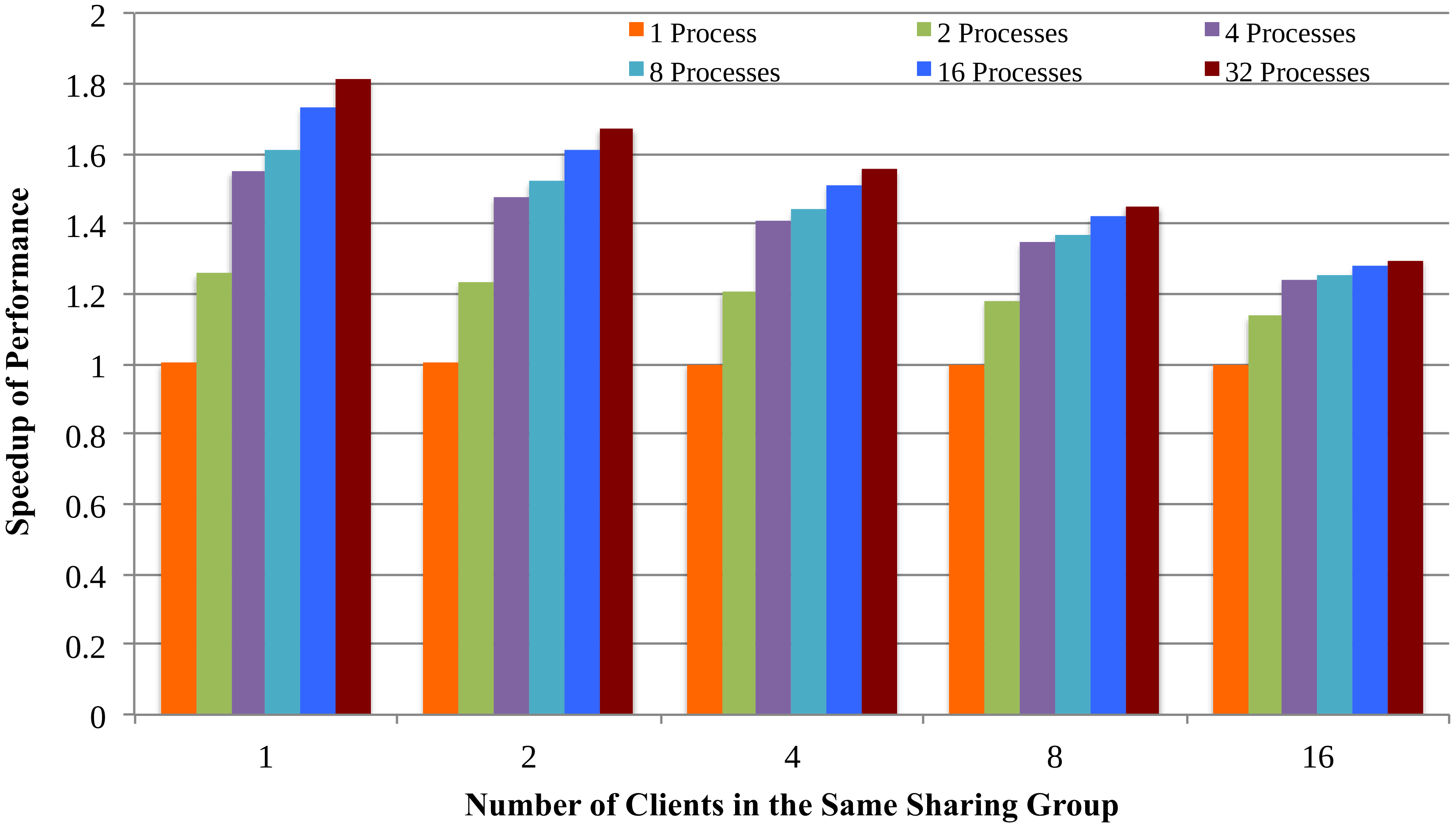
- With Scibox, data is merged before upload
- With Scibox, partial object access is supported

GTS Workload



	WSU	OSU	GT
GT	900 KB/s	4.4 MB/s	44 MB/s

Combustion Workload



- DR-function: (ImageName, DR8, FFT)
- 10, 000 images (~1.5 MB/image) shared via Scibox

Outline

- Background and Motivation
- Problems and Challenges
- Design and Implementation
- Evaluation
- Conclusion and Future Work

Conclusions and Future Work

Scibox: Cloud-based support for scientific data sharing

- Can operate across both public and private cloud stores

Conclusions and Future Work

Scibox: Cloud-based support for scientific data sharing

- Can operate across both public and private cloud stores

Data Reduction functions

- Exploit the structured nature of scientific data
- Reduce the amount of transferred data

Conclusions and Future Work

Scibox: Cloud-based support for scientific data sharing

- Can operate across both public and private cloud stores

Data Reduction functions

- Exploit the structured nature of scientific data
- Reduce the amount of transferred data

Transparent to Applications/Data Producers

- Implemented in ADIOS
- Can be also applied to other I/O libraries

Conclusions and Future Work

Scibox: Cloud-based support for scientific data sharing

- Can operate across both public and private cloud stores

Data Reduction functions

- Exploit the structured nature of scientific data
- Reduce the amount of transferred data

Transparent to Applications/Data Producers

- Implemented in ADIOS
- Can be also applied to other I/O libraries

Future Work

- Deploy Scibox on national labs' facilities to better understand potential use cases
- Additional optimizations of cloud storage for scientific data management

Thanks!