SGCI RESOURCE INVENTORY

Joe Stubbs, Suresh Marru, Steve Black, Dimuthu Wannipurage, D

Dec 06, 2021

CONTENTS

INTRODUCTION TO SCGI INVENTORY

1.1 Version

This is version 1.0.0 of the SGCI Resource Description Specification schema. This work is released under an Apache 2.0 license .

1.2 Introduction

The user-facing components of the Cyberinfrastructure (CI) ecosystem, science gateways and scientific workflow systems, share a common need of interfacing with physical resources (storage systems and execution environments) to manage data and execute codes (applications).

However, there is no uniform, platform-independent way to describe either the resources or the applications. To address this, we propose uniform semantics for describing resources and applications that will be relevant to a diverse set of stakeholders.

The SGCI Resource Description Specification provides a standard way for institutions and service providers to describe storage and computing infrastructure broadly available to the research computing and science gateway community. SGCI Resource descriptions provide a foundation for interoperability across gateway components and other cyberinfrastructure software.

The current, initial version of the resource description language focuses on "traditional" HPC and high-throughput storage and computing resources

1.3 Definitions

Definitions of terms used in the specification will be added here.

1.4 Specification Format

SGCI resource descriptions are JSON documents that conform to the JSONSchema definition describing a particular version of the SGCI Resource Description Specification.

1.5 Examples

We illustrate the main features of the specification by walking through a few prototypical examples.

SCIGAP Development Storage

A server or virtual machine providing storage accessible over SSH can be registered as resource with a single object provided within the `"storageResources"` attribute describing the connection information and the file systems present. In the SGCI Resource Descriptions specification, it is assumed that all resources provide some kind of storage capability; that is, at least one object within the `storageResources` array attribute must be provided, and within that object, at least one `connections` object must be provided.

A fundamental principle in the SGCI Resource Description Specification is that the `host` attribute uniquely identifies a resource, and only one description document for a given `host` can exist in the inventory. The value of `host` is a network addressable identifier for the resource, most typically, a fully qualified domain name.

The following example describes a hypothetical storage resource used by the SCIGAP framework in its development environment.

```
{
    "schemaVersion": "1.0.0",
    "name": "SCIGAP Development Storage",
    "host": "pgadev.scigap.org",
    "description": "POSIX storage server for the SCIGAP development environment.",
    "storageResources": [{
      "storageType": "POSIX",
      "connections": [{
          "connectionProtocol": "SSH",
          "securityProtocol": "SSHKEYS",
          "port": 22
    }],
      "fileSystems":
        [{
          "rootDir": "/"
        }]
    }]
}
```

Corral Storage System at TACC

The Corral storage system at TACC provides a more complicated example, with multiple file systems mounted onto a single resource and multiple types of storage endpoints available. The following example illustrates how a system like Corral, with many storage functionalities, can be described in a single document using the specification.

```
{
  "schemaVersion": "1.0.0",
  "host": "data.tacc.utexas.edu",
  "name": "tacc-corral-storage-login",
  "description": "TACC Corral high-performance storage resource",
  "storageResources": [
    {
      "storageType": "POSIX",
      "connections": [
        {
          "connectionProtocol": "SSH",
```

```
        "securityProtocol": "SSHKEYS",
        "port": 22
      },
      {
        "connectionProtocol": "SSH",
        "securityProtocol": "PASSWORDS",
        "port": 22
      },
      {
        "connectionProtocol": "SCP",
        "securityProtocol": "SSHKEYS",
        "port": 22
      },
      {
        "connectionProtocol": "SCP",
        "securityProtocol": "PASSWORDS",
        "port": 22
      },
      {
        "connectionProtocol": "SFTP",
        "securityProtocol": "SSHKEYS",
        "port": 22
      },
      {
        "connectionProtocol": "SFTP",
        "securityProtocol": "PASSWORDS",
        "port": 22
      }
    ],
    "fileSystems": [
      {
        "mountDir": "/home",
        "capacity": {
          "totalBytes": 940686700544
        }
      },
      {
        "mountDir": "/work",
        "capacity": {
          "totalBytes": 2040109484313600
        }
      }
    ]
  },
  {
    "storageType": "S3",
    "connections": [
      {
        "connectionProtocol": "HTTPS",
        "securityProtocol": "APIKEYS"
      }
    ],
```

```
      "fileSystems": []
    },
    {
      "storageType": "IRODS",
      "connections": [
        {
          "connectionProtocol": "IRODS",
          "securityProtocol": "PASSWORDS"
        }
      ],
      "fileSystems": []
    }
  ]
}
```

Carbonate HPC

Compute capabilities provided by resources are described within one or more `computeResources` definitions. Unlike the `storageResources` attribute that must contain at least one object, the `computeResources` attribute is entirely optional. Each compute resource object must define at least one `connections` object, analogous to the `storageResource` definitions. Additionally, each compute resource defines the way workloads are scheduled on the resource using the `schedulerType` property, with values such as `FORK` or `BATCH`. The value of `schedulerType` dictates additional objects that may be provided, such as the `batchSystem` object for value `BATCH`.

Carbonate is Indiana University's large-memory computer cluster. The simple description below only includes the `BATCH` submission capability and does not provide any partion (queue) information.

```
{
    "schemaVersion": "1.0.0",
    "name": "Carbonate HPC",
    "host": "carbonate.uits.iu.edu",
    "computeResources": [{
      "schedulerType": "BATCH",
      "connections": [{
          "connectionProtocol": "SSH",
          "securityProtocol": "SSHKEYS",
          "port": 22
      }],
      "batchSystem": {
          "jobManager": "SLURM",
          "commandPaths": [{
            "name": "SUBMISSION",
            "path": "/foo"
          }]
      }
    }],

    "storageResources": [{
      "storageType": "POSIX",
      "connections": [{
          "connectionProtocol": "SSH",
          "securityProtocol": "SSHKEYS",
          "port": 22
```

```
    }],

      "fileSystems": [{
          "rootDir": "/",
          "scratchDir": "/scratch"
      }]
    }]
}
```

TACC Stampede2 Cluster

In the final example describing the TACC Stampede2 supercomputer, two `computeResources` definitions are included, one with `schedulerType` having value `BATCH` and one with `schedulerType` having value `FORK`. While all "real" workloads are required to be submitted to the batch scheduler, the `FORK` scheduler could be utilized for "code compilation" tasks that run directly on the login node. Additionally, the `BATCH` compute resource includes descriptions of the partitions (queues). These are optional but very valuable for science gateway projects.

```
{
  "schemaVersion": "1.0.0",
  "host": "stampede2.tacc.xsede.org",
  "name": "tacc-xsede-stampede2",
  "description": "WIP: Sample resource for TACC Stampede2 Cluster",
  "computeResources": [
    {
      "schedulerType": "BATCH",
      "connections": [
        {
          "connectionProtocol": "SSH",
          "securityProtocol": "SSHKEYS",
          "port": 22
        },
        {
          "connectionProtocol": "SSH",
          "securityProtocol": "PASSWORDS",
          "port": 22
        }
      ],
      "batchSystem": {
        "jobManager": "SLURM",
        "commandPaths": [
          {
            "name": "SUBMISSION",
            "path": "/bin/sbatch"
          }
        ],
        "partitions": [
          {
            "name": "normal",
            "totalNodes": 256,
            "nodeHardware": {
              "cpuType": "KNL CPUs @ 1.40GHz",
              "cpuCount": 68,
              "memoryType": "DDR4",
```

```
          "memorySize": "96 GB"
        }
      }
    ],
    "executionCommands" : [
      {
        "commandType" : "mpi",
        "commandPrefix" : "ibrun",
        "moduleDependencies" : ["intel/17.0.4", "impi/17.0.3"]
      }
    ]
  }
},
{
  "schedulerType": "FORK",
  "connections": [
    {
      "connectionProtocol": "SSH",
      "securityProtocol": "SSHKEYS",
      "port": 22
    },
    {
      "connectionProtocol": "SSH",
      "securityProtocol": "PASSWORDS",
      "port": 22
    }
  ],
  "forkSystem": {
    "systemType": "LINUX"
  }
}
],
"storageResources": [
  {
    "storageType": "POSIX",
    "connections": [
      {
        "connectionProtocol": "SSH",
        "securityProtocol": "SSHKEYS",
        "port": 22
      }
    ],
    "fileSystems": [
      {
        "homeDir": "/home1",
        "scratchDir": "/scratch",
        "workDir": "/work"
      }
    ]
  }
]
}
```

1.6 Information Not In The Spec

Over the course of working on the specification, the project has decided to not include different resource types and/or additional attributes of existing resource types in version 1.0 for various reasons. In some cases, we plan to include the information in a subsequent version of the specification. In this section we collect some of the information not chosen for v 1.0, together with the rationale for not including it.

- Multi-factor authentication (MFA) requirements of a resource – MFA requirements are definitely important to capture and will be included in a subsequent version of the spec, but we want to make sure we have a good, general-purpose way of describing what seem to be a rapidly evolving aspect of cyberinfrastructure.

- Available Software Modules and other software on an execution system – The project may include such information in a subsequent version of the spec, but we want to make sure we have good mechanisms in place for working with information that is changing rapidly.

- More recent cloud storage and computing resource types – A number of different cloud resource types, including container orchestration systems such as Kubernetes and Docker Swarm, as well as Functions-as-a-service such as AWS Lambda, were intentionally not included in version 1.0. We hope to include these in a future release once the community has built some expertise incorporating them into science gateways.

- Databases and Web Services – Databases, including MySQL, Postgres, MongoDB, etc., and web service APIs like Figshare, Google Drive, etc., are not included in the spec, as these represent a significant departure from the kinds of resources defined in the current version. We do not currently plan to include them in any future version.

1.7 Integration
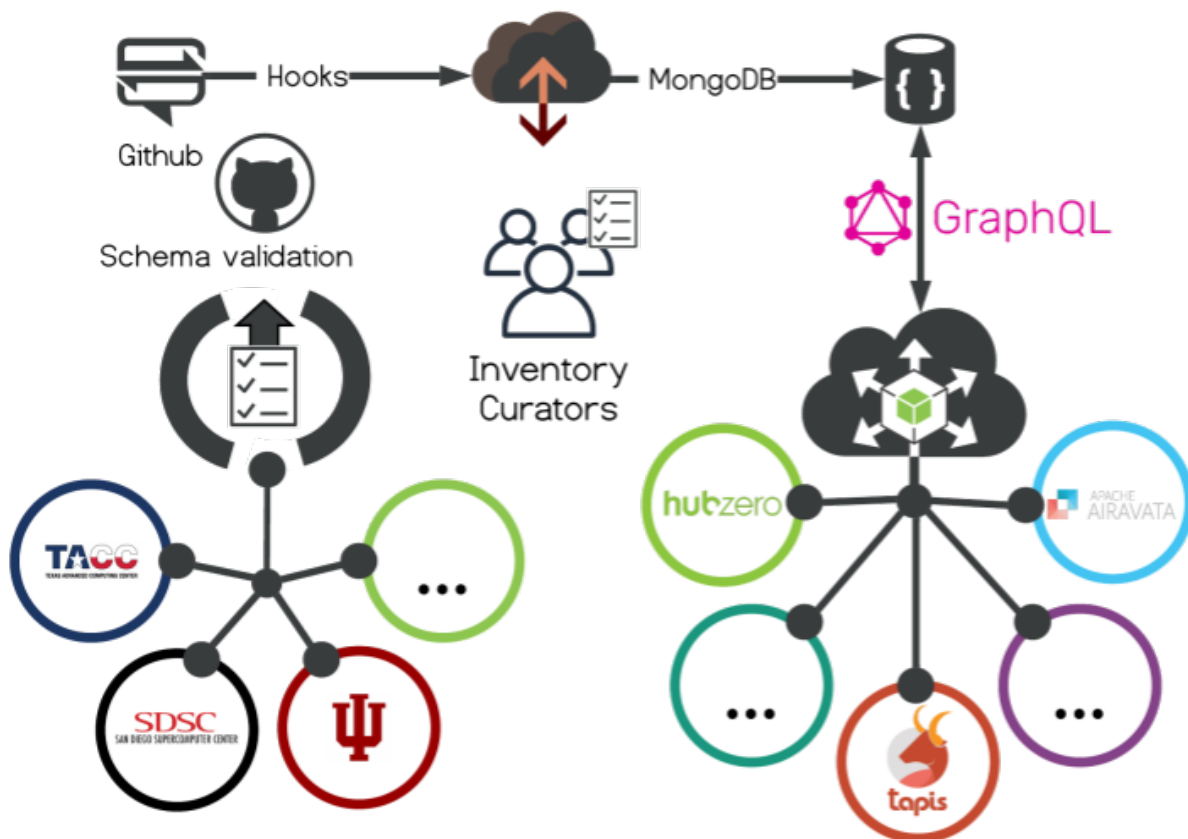
The SCGI Inventory is currently been integrated with Airavata, HUBzero ® , and Tapis. We expect the inventory to be adapted by others soon.

Links:

https://github.com/SGCI/sgci-resource-inventory

https://github.com/SGCI/sgci-resource-inventory-cache-service

Get Involved!

Issues, Comments, PRs all welcome!

SGCI: help@sciencegateways.org

Email: jstubbs at tacc.utexas.edu, smarru at iu.edu, dmejiapa at purdue.edu

STORAGE RESOURCE OBJECT

The `storageDefinitionList` attribute is made up of 1 or more `storageDefinition` objects. Each `storageDefinition` describes properties for interacting with one storage capability the resource provides. All resources in the SGCI Resource Catalog are assumed to provide storage facilities, and as such, all resource descriptions must include at least one `storageDefinition` object within the `storageDefinitionList` attribute.

The storage object is used to describe storage properties of the resource. All resources in the SGCI Resource Catalog are assumed to provide storage facilities, and as such, All resource descriptions must include values for the storage properties.			
type	*object*		
properties			
• **storageType**	High-level category to which this storage resource belongs. Determines additional properties that can be provided.		
	type	*string*	
	enum	POSIX, S3, IRODS	
• **connections**	The connection objects indicate available methods for accessing the resource.		
	type	*array*	
	items	#/definitions/connectionDefinition	
	minItems	1	
• **fileSystems**	The fileSystem objects containing information about the paths available on the storage resource.		
	type	*array*	
	items	#/definitions/fileSystemDefinition	
• defaultQuota	The defaultQuota object includes quota information about the available storage per		
	type	*object*	
	properties		
	• bytesPerUser	type	*integer*

2.1 Connection Object / connectionDefinition

The connection object defines all parameters required to establish a connection with the resource		
type	*object*	
properties		
• **connectionProtocol**	Communication protocol required to establish a connection with the resource	
	type	*string*
	enum	SSH, GLOBUS, HTTP, HTTPS, SFTP, SCP, IRODS
• **securityProtocol**	Cryptographic or access protocol enabled to access the resource	
	type	*string*
	enum	PASSWORDS, SSHKEYS, APIKEYS, X509, OAUTH2
• host	Communication endpoint host (if omitted, the top level host of this resource is used)	
	#/definitions/hostName	
• port	Communication endpoint port	
	type	*integer*
• proxyHost	The ProxyHost object includes details required to connect to the proxy tunneling	
	#/definitions/hostName	
• proxyPort	Communication endpoint required by the proxy	
	type	*integer*

2.2 File System Object / fileSystemDefinition

The fileSystem object contains information about the paths available on the storage resource.		
type	*object*	
properties		
• mountDir	The path on the resource where the fileSystem is mounted.	
	#/definitions/dirPath	
• homeDir	The path on the resource serving as the base for user specific home directories.	
	#/definitions/dirPath	
• scratchDir	The path on the resource serving as the base for the SCRATCH file system, typically non-shared storage where running jobs should direct their I/O while executing.	
	#/definitions/dirPath	
• workDir	The path on the resource serving as the base for the WORK file system, typically shared storage where files are staged before being copied to SCRATCH.	
	#/definitions/dirPath	
• capacity	The capacity object describes the total available storage on the fileSystem.	
	type	*object*
	properties	
	• totalBytes	The total bytes of storage available on the fileSystem.
		type	*integer*

2.3 Directory Path Object / dirPath

the general form of the name of a file or directory, it specifies a unique location in the file system	
type	*string*
maxLength	1024
minLength	1

2.4 Examples

SCIGAP Development Storage

```
{
    "schemaVersion": "1.0.0",
    "name": "SCIGAP Development Storage",
    "host": "pgadev.scigap.org",
    "description": "POSIX storage server for the SCIGAP development environment.",
    "storageResources": [{
      "storageType": "POSIX",
      "connections": [{
          "connectionProtocol": "SSH",
          "securityProtocol": "SSHKEYS",
          "port": 22
    }],
      "fileSystems":
        [{
          "rootDir": "/"
        }]
    }]
}
```

COMPUTE RESOURCE OBJECT

The Compute Resource Object is used to describe computational properties of the resource. Not all resources in the SGCI Resource Catalog are assumed to provide compute facilities.		
type	*object*	
properties		
• schedulerType	Job scheduler configured on the resource	
	type	*string*
	enum	FORK, BATCH
• connections	The connection objects indicate available methods for accessing the resource.	
	type	*array*
	items	#/definitions/connectionDefinition
	minItems	1
• executionCommands	This list defines all the commands required to execute any command on the resource	
	type	*array*
	items	#/definitions/executionCommandDefinition
• batchSystem	The batchSystem object lists all parameters required by the batch scheduler system	
	#/definitions/batchSystemDefinition	
• forkSystem	The forkSystem object lists all parameters required by the fork scheduler system	
	#/definitions/forkSystemDefinition	

3.1 Connection Object / connectionDefinition

The connection object defines all parameters required to establish a connection with the resource		
type	*object*	
properties		
• **connectionProtocol**	Communication protocol required to establish a connection with the resource	
	type	*string*
	enum	SSH, GLOBUS, HTTP, HTTPS, SFTP, SCP, IRODS
• **securityProtocol**	Cryptographic or access protocol enabled to access the resource	
	type	*string*
	enum	PASSWORDS, SSHKEYS, APIKEYS, X509, OAUTH2
• host	Communication endpoint host (if omitted, the top level host of this resource is used)	
	#/definitions/hostName	
• port	Communication endpoint port	
	type	*integer*
• proxyHost	The ProxyHost object includes details required to connect to the proxy tunneling	
	#/definitions/hostName	
• proxyPort	Communication endpoint required by the proxy	
	type	*integer*

3.2 Batch System Object / batchSystemDefinition

type	*object*	
properties		
• **jobManager**	Job scheduler software installed on the resource no handle job requests	
	type	*string*
	enum	SLURM, SGE, PBS, PBSPRO, TORQUE, LOADLEVELER, LSF
• host	Hostname of the job manager	
	#/definitions/hostName	
• commandPaths	List of commands supported by the Batch System	
	type	*array*
	items	#/definitions/commandPathDefinition
• partitions	List of public partitions and queues installed on the Batch System	
	type	*array*
	items	#/definitions/partitionDefinition

3.3 fork System Object / forkSystemDefinition

type	*object*	
properties		
• **systemType**	OS installed on the resource	
	type	*string*
	enum	LINUX, WINDOWS
• version	Version of OS installed on the resource	
	type	*string*
• nodeHardware	NodeHardware object contains details about the software installed on the resource	
	#/definitions/nodeHardwareDefinition	

3.4 Examples

Carbonate HPC Cluster

```
{
    "schemaVersion": "1.0.0",
    "name": "Carbonate HPC",
    "host": "carbonate.uits.iu.edu",
    "computeResources": [{
      "schedulerType": "BATCH",
      "connections": [{
          "connectionProtocol": "SSH",
          "securityProtocol": "SSHKEYS",
          "port": 22
      }],
      "batchSystem": {
          "jobManager": "SLURM",
          "commandPaths": [{
            "name": "SUBMISSION",
            "path": "/foo"
          }]
      }
    }],

    "storageResources": [{
      "storageType": "POSIX",
      "connections": [{
          "connectionProtocol": "SSH",
          "securityProtocol": "SSHKEYS",
          "port": 22
    }],

      "fileSystems": [{
          "rootDir": "/",
          "scratchDir": "/scratch"
      }]
    }]
```

```
}
```

TACC Stampede2 Cluster

```
{
  "schemaVersion": "1.0.0",
  "host": "stampede2.tacc.xsede.org",
  "name": "tacc-xsede-stampede2",
  "description": "WIP: Sample resource for TACC Stampede2 Cluster",
  "computeResources": [
    {
      "schedulerType": "BATCH",
      "connections": [
        {
          "connectionProtocol": "SSH",
          "securityProtocol": "SSHKEYS",
          "port": 22
        },
        {
          "connectionProtocol": "SSH",
          "securityProtocol": "PASSWORDS",
          "port": 22
        }
      ],
      "batchSystem": {
        "jobManager": "SLURM",
        "commandPaths": [
          {
            "name": "SUBMISSION",
            "path": "/bin/sbatch"
          }
        ],
        "partitions": [
          {
            "name": "normal",
            "totalNodes": 256,
            "nodeHardware": {
              "cpuType": "KNL CPUs @ 1.40GHz",
              "cpuCount": 68,
              "memoryType": "DDR4",
              "memorySize": "96 GB"
            }
          }
        ],
        "executionCommands" : [
          {
            "commandType" : "mpi",
            "commandPrefix" : "ibrun",
            "moduleDependencies" : ["intel/17.0.4", "impi/17.0.3"]
          }
        ]
      }
    }
```

```
    },
    {
      "schedulerType": "FORK",
      "connections": [
        {
          "connectionProtocol": "SSH",
          "securityProtocol": "SSHKEYS",
          "port": 22
        },
        {
          "connectionProtocol": "SSH",
          "securityProtocol": "PASSWORDS",
          "port": 22
        }
      ],
      "forkSystem": {
        "systemType": "LINUX"
      }
    }
  ],
  "storageResources": [
    {
      "storageType": "POSIX",
      "connections": [
        {
          "connectionProtocol": "SSH",
          "securityProtocol": "SSHKEYS",
          "port": 22
        }
      ],
      "fileSystems": [
        {
          "homeDir": "/home1",
          "scratchDir": "/scratch",
          "workDir": "/work"
        }
      ]
    }
  ]
}
```

FOUR

STATUS OBJECT

4.1 Examples

XSEDE Comet Resource

```
{
  "sgciResources": [
    {
      "schemaVersion":"0.1.0",
      "host":"comet.sdsc.xsede.org",
      "name":"SDSC Dell Cluster with Intel Haswell Processors (Comet)",
      "description":"Comet is a 2.0 Petaflop (PF) Dell integrated compute cluster, with␣
→next-generation Intel Haswell processors (with AVX2), interconnected with Mellanox FDR␣
→InfiniBand in a hybrid fat-tree topology. Full bisection bandwidth is available at␣
→rack level (72 nodes) and there is a 4:1 oversubscription cross-rack. Compute nodes␣
→feature 320 GB of SSD storage and 128GB of DRAM per node. The system also features 7PB␣
→of performance storage (200GB/s aggregate), and 6PB of durable storage. Additionally,␣
→four 1.5TB large memory nodes and additional nodes for Gateway hosting and VM image␣
→repositories are available. Comet will enable high performance virtualization using␣
→the single root I/O virtualization (SR-IOV) technology.",
      "computeResource": [
        {
          "schedulerType":"BATCH",
          "connections": [
            {
              "connectionProtocol":"GRIDFTP",
              "securityProtocol":"X509",
              "proxyHost":"oasis-dm.sdsc.xsede.org",
              "proxyPort":2811
            },
            {
              "connectionProtocol":"SSH",
              "securityProtocol":"SSHKEYS",
              "port":22
            },
            {
              "connectionProtocol":"SSH",
              "securityProtocol":"X509",
              "port":22
            },
```

```
                    {
                       "connectionProtocol":"SCP",
                       "securityProtocol":"SSHKEYS",
                       "port":22
                    },
                    {
                       "connectionProtocol":"SCP",
                       "securityProtocol":"X509",
                       "port":22
                    }
                 ],
                 "batchSystem":
                    {
                       "jobManager":""
                    }
              },
              {
                 "schedulerType":"FORK",
                 "forkSystem":
                    {
                       "systemType":"LINUX"
                    },
                 "connections": [
                    {
                       "connectionProtocol":"GRIDFTP",
                       "securityProtocol":"X509",
                       "proxyHost":"oasis-dm.sdsc.xsede.org",
                       "proxyPort":2811
                    },
                    {
                       "connectionProtocol":"SSH",
                       "securityProtocol":"SSHKEYS",
                       "port":22
                    },
                    {
                       "connectionProtocol":"SSH",
                       "securityProtocol":"X509",
                       "port":22
                    },
                    {
                       "connectionProtocol":"SCP",
                       "securityProtocol":"SSHKEYS",
                       "port":22
                    },
                    {
                       "connectionProtocol":"SCP",
                       "securityProtocol":"X509",
                       "port":22
                    }
                 ]
              }
           ],
```

```
      "resourceStatus":
        {
          "status":"Production",
          "starts":"2015-04-01",
          "ends":"2021-03-31"
        },
      "resourceOutages": [
        {
          "type":"Full",
          "name":"SDSC Comet Lustre projects filesystem issue",
          "description":"We are currently seeing problems with one of the Comet Lustre
→projects filesystem servers. This has resulted in some of the Lustre storage targets
→going offline. There is a reservation in place to prevent new jobs from starting and
→we will update once we have more info on the timeline for resolution of the problem.
→Please email help@xsede.org if you have any questions.",
          "url":"https://www.xsede.org/news/-/news/item/12691",
          "starts":"2020-11-26T02:00:00Z",
          "ends":"2020-11-28T02:00:00Z"
        }
      ]
    }
  ]
}
```

FIVE

OUTAGE OBJECT

The `outageDefinitionList` attribute is made up of 0 or more `outageDefinition` objects. Each `outageDefinition` describes properties for one current or future outage affecting the resource.

type	*object*	
properties		
• outageType	It the outage full or partial	
	type	*string*
	enum	Partial, Full
• name	Outage descriptive name	
	type	*string*
	maxLength	128
	minLength	1
• description	Outage description	
	type	*string*
	maxLength	4096
	minLength	1
• url	Outage details URL	
	type	*string*
	format	uri
• **startsDatetime**	Starting date and time in UTC and ISO 8601 format	
	type	*string*
	format	date-time
• **endsDatetime**	Ending date and time in UTC and ISO 8601 format	
	type	*string*
	format	date-time

5.1 Examples

XSEDE Comet Resource

```
{
  "sgciResources": [
    {
      "schemaVersion":"0.1.0",
      "host":"comet.sdsc.xsede.org",
      "name":"SDSC Dell Cluster with Intel Haswell Processors (Comet)",
      "description":"Comet is a 2.0 Petaflop (PF) Dell integrated compute cluster, with
→next-generation Intel Haswell processors (with AVX2), interconnected with Mellanox FDR
→InfiniBand in a hybrid fat-tree topology. Full bisection bandwidth is available at
→rack level (72 nodes) and there is a 4:1 oversubscription cross-rack. Compute nodes
→feature 320 GB of SSD storage and 128GB of DRAM per node. The system also features 7PB
→of performance storage (200GB/s aggregate), and 6PB of durable storage. Additionally,
→four 1.5TB large memory nodes and additional nodes for Gateway hosting and VM image
→repositories are available. Comet will enable high performance virtualization using
→the single root I/O virtualization (SR-IOV) technology.",
```

```
    "computeResource": [
      {
        "schedulerType":"BATCH",
        "connections": [
          {
            "connectionProtocol":"GRIDFTP",
            "securityProtocol":"X509",
            "proxyHost":"oasis-dm.sdsc.xsede.org",
            "proxyPort":2811
          },
          {
            "connectionProtocol":"SSH",
            "securityProtocol":"SSHKEYS",
            "port":22
          },
          {
            "connectionProtocol":"SSH",
            "securityProtocol":"X509",
            "port":22
          },
          {
            "connectionProtocol":"SCP",
            "securityProtocol":"SSHKEYS",
            "port":22
          },
          {
            "connectionProtocol":"SCP",
            "securityProtocol":"X509",
            "port":22
          }
        ],
        "batchSystem":
          {
            "jobManager":""
          }
      },
      {
        "schedulerType":"FORK",
        "forkSystem":
          {
            "systemType":"LINUX"
          },
        "connections": [
          {
            "connectionProtocol":"GRIDFTP",
            "securityProtocol":"X509",
            "proxyHost":"oasis-dm.sdsc.xsede.org",
            "proxyPort":2811
          },
          {
            "connectionProtocol":"SSH",
            "securityProtocol":"SSHKEYS",
```

```
              "port":22
            },
            {
              "connectionProtocol":"SSH",
              "securityProtocol":"X509",
              "port":22
            },
            {
              "connectionProtocol":"SCP",
              "securityProtocol":"SSHKEYS",
              "port":22
            },
            {
              "connectionProtocol":"SCP",
              "securityProtocol":"X509",
              "port":22
            }
          ]
        }
      ],
      "resourceStatus":
        {
          "status":"Production",
          "starts":"2015-04-01",
          "ends":"2021-03-31"
        },
      "resourceOutages": [
        {
          "type":"Full",
          "name":"SDSC Comet Lustre projects filesystem issue",
          "description":"We are currently seeing problems with one of the Comet Lustre␣
→projects filesystem servers. This has resulted in some of the Lustre storage targets␣
→going offline. There is a reservation in place to prevent new jobs from starting and␣
→we will update once we have more info on the timeline for resolution of the problem.␣
→Please email help@xsede.org if you have any questions.",
          "url":"https://www.xsede.org/news/-/news/item/12691",
          "starts":"2020-11-26T02:00:00Z",
          "ends":"2020-11-28T02:00:00Z"
        }
      ]
    }
  ]
}
```

COMPLETE INVENTORY SCHEMA

http://sciencegateways.org/SGCIResource		
Schema for an SGCI Resource		
type	*object*	
properties		
• **schemaVersion**	Version of the SGCI Resource Descriptions schema used for this description.	
	type	*string*
• **host**	Network addressable name (hostname) serving as a unique identifier for the resource across all SGCI resource descriptions.	
	hostName	
• **name**	Human-readable, displayable name for the resource	
	resourceName	
• description	Detailed description of the resource.	
	type	*string*
	maxLength	2048
• **storageResources**	*storageDefinitionList*	
• computeResources	*computeDefinitionList*	
• resourceStatus	*statusDefinition*	
• resourceOutages	*outageDefinitionList*	
additionalProperties	False	

6.1 resourceName

Human-readable, displayable name for the resource	
type	*string*
maxLength	256
minLength	1

6.2 hostName

label assigned to identify the device as an internet point of access	
type	*string*
maxLength	256
minLength	1

6.3 dirPath

the general form of the name of a file or directory, it specifies a unique location in the file system	
type	*string*
maxLength	1024
minLength	1

6.4 connectionDefinition

The connection object defines all parameters required to establish a connection with the resource		
type	*object*	
properties		
• **connectionProtocol**	Communication protocol required to establish a connection with the resource	
	type	*string*
	enum	SSH, GLOBUS, HTTP, HTTPS, SFTP, SCP, IRODS
• **securityProtocol**	Cryptographic or access protocol enabled to access the resource	
	type	*string*
	enum	PASSWORDS, SSHKEYS, APIKEYS, X509, OAUTH2
• host	Communication endpoint host (if omitted, the top level host of this resource is used)	
	hostName	
• port	Communication endpoint port	
	type	*integer*
• proxyHost	The ProxyHost object includes details required to connect to the proxy tunneling	
	hostName	
• proxyPort	Communication endpoint required by the proxy	
	type	*integer*

6.5 storageDefinitionList

type	*array*
items	*storageDefinition*

6.6 computeDefinitionList

type	*array*
items	*computeDefinition*

6.7 outageDefinitionList

type	*array*
items	*outageDefinition*

6.8 storageDefinition

<table>
<tr><td colspan="4">The storage object is used to describe storage properties of the resource. All resources in the SGCI Resource Catalog are assumed to provide storage facilities, and as such, All resource descriptions must include values for the storage properties.</td></tr>
<tr><td colspan="2">type</td><td colspan="2">object</td></tr>
<tr><td colspan="4">properties</td></tr>
<tr><td>• storageType</td><td colspan="3">High-level category to which this storage resource belongs. Determines additional properties that can be provided.</td></tr>
<tr><td></td><td>type</td><td colspan="2">string</td></tr>
<tr><td></td><td>enum</td><td colspan="2">POSIX, S3, IRODS</td></tr>
<tr><td>• connections</td><td colspan="3">The connection objects indicate available methods for accessing the resource.</td></tr>
<tr><td></td><td>type</td><td colspan="2">array</td></tr>
<tr><td></td><td>items</td><td colspan="2">connectionDefinition</td></tr>
<tr><td></td><td>minItems</td><td colspan="2">1</td></tr>
<tr><td>• fileSystems</td><td colspan="3">The fileSystem objects containing information about the paths available on the storage resource.</td></tr>
<tr><td></td><td>type</td><td colspan="2">array</td></tr>
<tr><td></td><td>items</td><td colspan="2">fileSystemDefinition</td></tr>
<tr><td>• defaultQuota</td><td colspan="3">The defaultQuota object includes quota information about the available storage per</td></tr>
<tr><td></td><td>type</td><td colspan="2">object</td></tr>
<tr><td></td><td colspan="3">properties</td></tr>
<tr><td></td><td>• bytesPerUser</td><td>type</td><td>integer</td></tr>
</table>

6.9 fileSystemDefinition

The fileSystem object contains information about the paths available on the storage resource.			
type	*object*		
properties			
• mountDir	The path on the resource where the fileSystem is mounted.		
	dirPath		
• homeDir	The path on the resource serving as the base for user specific home directories.		
	dirPath		
• scratchDir	The path on the resource serving as the base for the SCRATCH file system, typically non-shared storage where running jobs should direct their I/O while executing.		
	dirPath		
• workDir	The path on the resource serving as the base for the WORK file system, typically shared storage where files are staged before being copied to SCRATCH.		
	dirPath		
• capacity	The capacity object describes the total available storage on the fileSystem.		
	type	*object*	
	properties		
	• totalBytes	The total bytes of storage available on the fileSystem.	
		type	*integer*

6.10 computeDefinition

The Compute Resource Object is used to describe computational properties of the resource. Not all resources in the SGCI Resource Catalog are assumed to provide compute facilities.		
type	*object*	
properties		
• schedulerType	Job scheduler configured on the resource	
	type	*string*
	enum	FORK, BATCH
• connections	The connection objects indicate available methods for accessing the resource.	
	type	*array*
	items	*connectionDefinition*
	minItems	1
• executionCommands	This list defines all the commands required to execute any command on the resource	
	type	*array*
	items	*executionCommandDefinition*
• batchSystem	The batchSystem object lists all parameters required by the batch scheduler system	
	batchSystemDefinition	
• forkSystem	The forkSystem object lists all parameters required by the fork scheduler system	
	forkSystemDefinition	

6.11 executionCommandDefinition

The execution Command Object describes how a commnad should be executed on the resource			
type	*object*		
properties			
• **commandType**	label that describe the type of command supported by the resource		
	type	*string*	
	examples	serial	
		mpi	
		openmp	
		ccm	
• commandPrefix	command to be preappend to the command in order to be executed as the command-Type, e.g mpi->mpirun		
	type	*string*	
	examples	ibrun	
		mpirun	
• moduleDependen-cies	list of modules to be loaded before execution of the command		
	type	*array*	
	items	type	*string*

6.12 batchSystemDefinition

type	*object*		
properties			
• **jobManager**	Job scheduler software installed on the resource no handle job requests		
	type	*string*	
	enum	SLURM, SGE, PBS, PBSPRO, TORQUE, LOADLEVELER, LSF	
• host	Hostname of the job manager		
	hostName		
• commandPaths	List of commands supported by the Batch System		
	type	*array*	
	items	*commandPathDefinition*	
• partitions	List of public partitions and queues installed on the Batch System		
	type	*array*	
	items	*partitionDefinition*	

6.13 commandPathDefinition

type	*object*		
properties			
• **name**	Label that defines the operation supported by the system		
	type	*string*	
	examples	SUBMISSION	
		JOB_MONITORING	
		DELETION	
		CHECK_JOB	
		SHOW_QUEUE	
		SHOW_RESERVATION	
		SHOW_START	
• **path**	Complete path to the command that executes the operation		
	type	*string*	

6.14 partitionDefinition

type	*object*		
properties			
• **name**	This label represents a partition of hardware for the resource, typically a queue or partition in the job scheduler		
	type	*string*	
	examples	normal	
		large-mem	
• submitArgs	Arguments required to request this partition of hardware (if omitted, '-p PARTI-TION_NAME' or '-q PARTITION_NAME' is used, depending on the job scheduler)		
	type	*array*	
	examples	–nodes=1	
		–partition=hugemem	
		–exclusive	
		–constraint=40core	
	items	type	*string*
• totalNodes	Number of available nodes on this partition / queue		
	type	*integer*	
• nodeHardware	The nodeHardware object includes detailed information of the node		
	nodeHardwareDefinition		
• computeQuotas	queues or partition may have multiple restriction on the jobs allowed to run		
	type	*object*	
	properties		
	• minJobsTotal	Minimum number of allowed jobs actively running on the partition	
		type	*integer*
	• maxJobsTotal	Maximum number of allowed jobs actively running on the partition	
		type	*integer*
	• minJobsPerUser	Minimum number of allowed jobs actively running on the partition for a user	
		type	*integer*

Table 1 – continued from previous page

	• maxJobsPerUser	Maximum number of allowed jobs actively running on the partition for a user	
		type	*integer*
	• minNodesPerJob	Minimum number of nodes allowed to be used by a job	
		type	*integer*
	• maxNodesPerJob	Maximum number of nodes allowed to be used by a job	
		type	*integer*
	• minTimePerJob	Minimum time a job can run without been killed	
		type	*integer*
	• maxTimePerJob	Maximum time a job can run without been killed	
		type	*integer*
	• minMemoryPerJob	Minimum memory allocated for a job	
		type	*string*
	• maxMemoryPerJob	Maximum memory allocated for a job	
		type	*string*
	• minCPUsPerJob	Minimum number of cpus allowed to be used by a job	
		type	*integer*
	• maxCPUsPerJob	Maximum number of cpus allowed to be used by a job	
		type	*integer*
	• minGPUsPerJob	Minimum number of gpus allowed to be used by a job	
		type	*integer*
	• maxGPUsPerJob	Maximum number of gpus allowed to be used by a job	
		type	*integer*
	• minCPUsPerNode	Minimum number of cpus allowed to be requested for a node	
		type	*integer*
	• maxCPUsPerNode	Maximum number of cpus allowed to be requested for a node	
		type	*integer*

6.15 forkSystemDefinition

type	*object*	
properties		
• **systemType**	OS installed on the resource	
	type	*string*
	enum	LINUX, WINDOWS
• version	Version of OS installed on the resource	
	type	*string*
• nodeHardware	NodeHardware object contains details about the software installed on the resource	
	nodeHardwareDefinition	

6.16 nodeHardwareDefinition

type	*object*	
properties		
• cpuType	Type of cpus installed on the resource	
	type	*string*
	examples	Haswell CPUs @ 2.60GHz
		Intel Xeon-E5
• cpuCount	Total number of cpus available on the resource	
	type	*integer*
• gpuType	Type of gpus installed on the resource	
	type	*string*
	examples	NVIDIA Tesla P100
		NVIDIA Tesla M2090
• gpuCount	Total number of gpus available on the resource	
	type	*integer*
• memoryType	Type of memory installed on the resource	
	type	*string*
• memorySize	Total memory available on the resource	
	type	*string*
	examples	64 GB

6.17 statusDefinition

type	*object*	
properties		
• **status**	Descriptive current resource status	
	type	*string*
	enum	Pre-production, Production, Post-production
• startDate	When the resource starts the current status	
	type	*string*
	format	date
• endsDate	When the resource ends the current status	
	type	*string*
	format	date

6.18 outageDefinition

type	*object*	
properties		
• outageType	It the outage full or partial	
	type	*string*
	enum	Partial, Full
• name	Outage descriptive name	
	type	*string*
	maxLength	128
	minLength	1
• description	Outage description	
	type	*string*
	maxLength	4096
	minLength	1
• url	Outage details URL	
	type	*string*
	format	uri
• **startsDatetime**	Starting date and time in UTC and ISO 8601 format	
	type	*string*
	format	date-time
• **endsDatetime**	Ending date and time in UTC and ISO 8601 format	
	type	*string*
	format	date-time

INDICES AND TABLES

- search