www.neesgrid.org

(Last modified: May 13, 2003)

NEESgrid System Architecture

Version 1.1

Carl Kesselman[1] Randy Butler[2] Ian Foster[3,4] Joe Futrelle[2]
Doru Marcusiu[2] Sridhar Gulipalli[1] Laura Pearlman[1] Chuck Severance[5]

[1] University of Southern California Information Sciences Institute, Marina del Rey, CA 90292

[2] National Center for Supercomputing Applications, Urbana-Champaign, IL 61801

[3] Argonne National Laboratory, Argonne, IL 60439

[4] University of Chicago, Chicago, IL 60637

[5] University of Michigan, Ann Arbor, MI 48109

Feedback on this document should be directed to neesgrid-si@neesgrid.org

Acknowledgment: This work was supported primarily by the George E. Brown, Jr. Network for Earthquake Engineering Simulation (NEES) Program of the National Science Foundation under Award Number CMS-0117853.

Table of Contents

1 Introduction

This document describes the system architecture for the NEESgrid project, the system integration element of the George E. Brown, Jr, Network for Earthquake Engineering Simulation (NEES). The result of NEESgrid will be a collaboratory, or "laboratory without walls," for earthquake engineering research. As illustrated in Figure 1, the NEESgrid collaboratory will link the national earthquake engineering community with up to twenty laboratories (see Figure 2), allowing teleoperation and teleobservation of experimental apparatus via the Internet (collectively, "telepresence"), access to data produced by these apparatus, and hybrid operations integrating physical and numerical simulations.

When completed, the NEESgrid collaboratory will enable four core capabilities. First, the collaboratory will link scientists to facilities to allow remote operation and observation of experimental equipment. Second, the collaboratory will link scientists to data to allow remote, shared viewing of real-time and archival data. Third, the collaboratory will link facilities with data, not only in the form of data repositories but also in the form of output from numerical simulations joined to physical simulations. Finally, the collaboratory will link scientists with other scientists, independent of time and location, to allow joint editing of documents, joint viewing of data visualizations, and joint operation and observation of physical and numerical simulations.

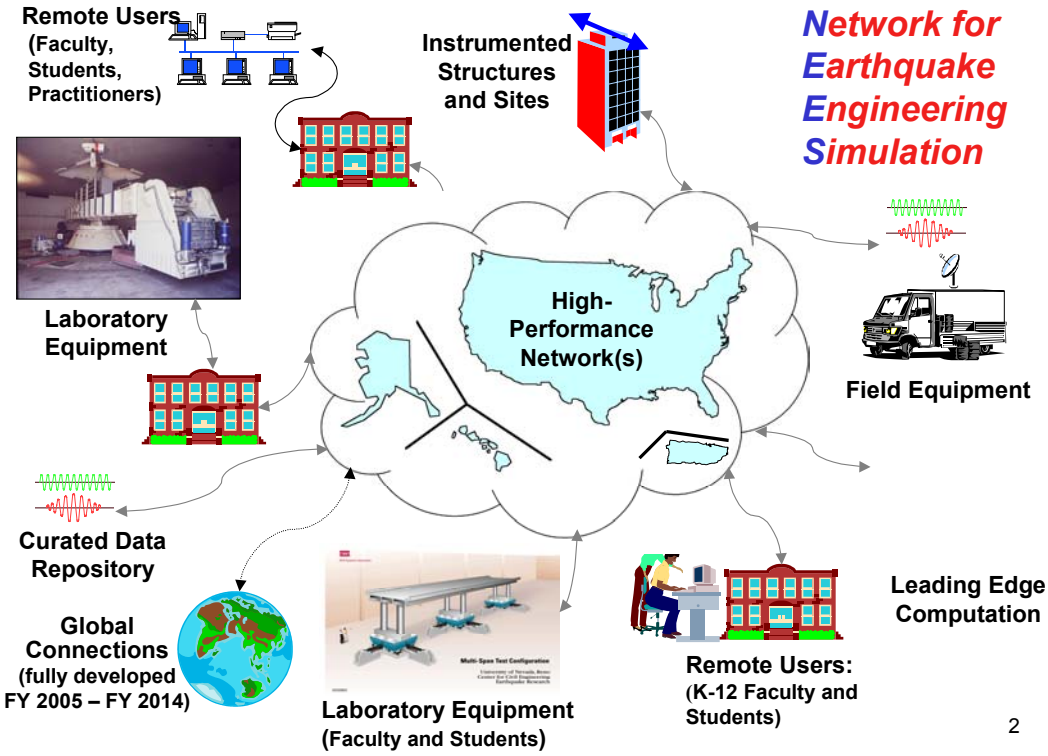

Figure 1: The NEESgrid concept

The NEESgrid system architecture specifies the software that will be put in place to enable these four core capabilities. This software comprises components intended for deployment at laboratories, at other resource sites, at a central operations site (initially, NCSA), and at user sites. Much of this software will be integrated (and, in some cases produced) by the NEESgrid team. In

important cases, however, the NEESgrid team will produce or modify harnesses (such as the CHEF collaborative framework) into which must be plugged site-supplied software to deal with, for example, specific equipment, data formats, data browsers, or collaboration tools.

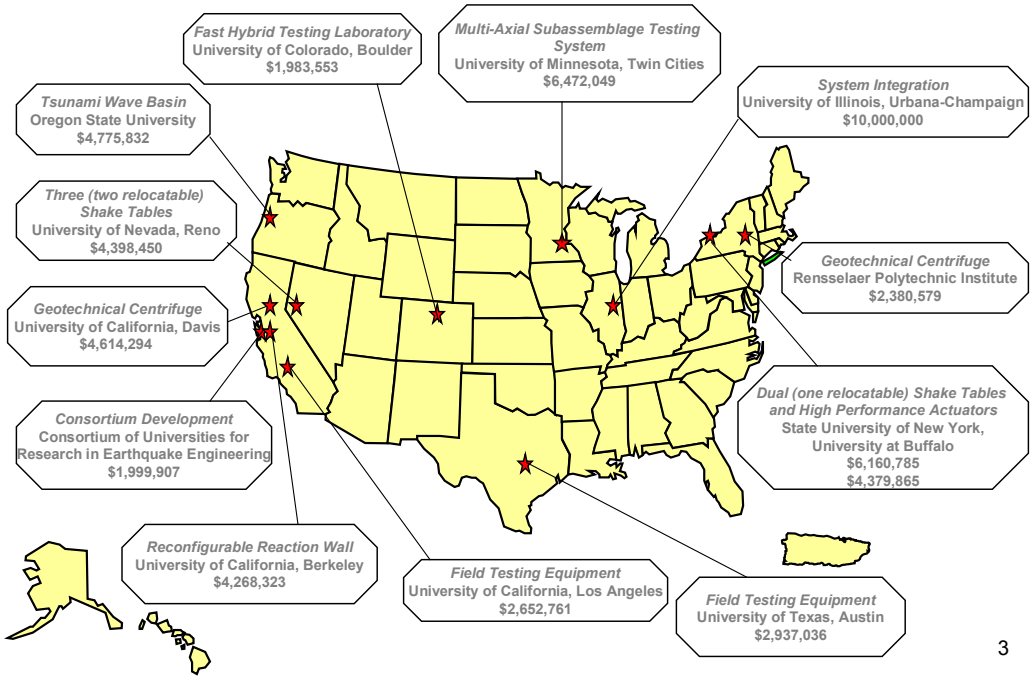

Figure 2: NEES awards to date

The NEESgrid system architecture design presented here is based on the results of the user requirements analysis performed by the NEESgrid User Requirements team [4]; additional site visits performed by the NEESgrid system architecture team; and the experience of the system architecture team in constructing similar systems. The development of the NEESgrid software, and inevitable associated refinements to the NEESgrid system architecture definition, will follow the frequently used spiral development model, in which simple capabilities are delivered early to gain user feedback, which then serves to guide further development. Thus, while in this document we specify the capabilities planned each of the three official releases of the NEESgrid software (NEESsoft-1, 2, and 3, planned for the end of year 1, 2, and 3), we expect that additional intermediate releases will occur, and we point out that the capabilities of later releases may change as a result of user experience and the ongoing requirements analysis process.

The NEESgrid software itself is to be constructed by the integration of

- a body of extant software, particularly significant elements of the NSF Middleware Initiative (NMI) software system, as produced by the GRIDS Center (www.grids-center.org),
- a small amount of custom software to be produced by the NEESgrid team to address general NEESgrid issues, and
- various site-specific, data-format-specific, and collaboration-tool-specific software to be produced by the equipment sites, other NEES participants, or other sources.

We take care in this document to specify clearly what software falls into each of these three classes.

2 User Requirements

The following text, taken from the NEESgrid User Requirements Document [4], summarizes our current understanding of NEES user requirements. We refer the reader to that document for more details on this topic.

As a result of preliminary interviewing, observation, and surveying within the earthquake engineering community, data suggest the following critical baseline use scenarios within the core collaboratory capabilities identified above:

1) *Linking scientists and facilities*:
   a) observation of an experiment in progress;
   b) observation before and after an experiment (e.g., placement of sensors within a specimen or assessment of damage to a specimen); and
   c) remote operation of an experiment—although support for teleoperation is less emphatic, with severe reservations about compromised safety because of control latency introduced via the Internet.
2) *Linking facilities and data*:
   a) hybrid operation of physical simulations with other simulations, both physical and numerical – noting again a high level of concern over problems associated with network delays; and
   b) automatic archiving of raw data, calibration data, and processed data—with some disagreement about the degree of local archiving vs. central archiving, and concerns about control over data by principal investigators.
3) *Linking scientists and data*:
   a) collaborative views of time synchronized data visualizations (both parameter/s vs. time and parameter vs. parameter—such as a hysteresis plot of force vs. displacement unfolding over a time interval), where the layout of these views, the contents, and the format are user-defined-and the same mechanism is used for real-time and for archival data; and
   b) collaborative views of time synchronized data visualizations with video and audio recordings, such as a hysteresis plot unfolding over a time interval with the corresponding views and sounds from cameras and microphones used to record an experiment—again, using the same mechanism to view real-time and archival data.
4) *Linking scientists and other scientists*:
   a) synchronous communication, such as with colleagues during an experiment—including video and data conferencing and shared teleobservation views; and
   b) asynchronous communication, such as with colleagues over the course of preparing a publication resulting from an experiment—including secure access to common files, tools for sharing comments, and tools for notifying collaborators about changes or additions to shared work objects, such as papers, datasets, and analyses.

3 Architecture Overview

We introduce the NEESgrid system architecture by reviewing briefly the physical elements that are to be connected, the capabilities that we seek to provide, and the principal software components that will be used to provide those capabilities.

3.1 Physical Elements

We conceive of NEESgrid as comprising the following physical elements (see Figure 1):

- A moderate number of *equipment sites*, at which are located experimental facilities such as shake tables, centrifuges, and wave tanks—as well as, eventually, mobile facilities.
- A moderate number of *resource sites*, at which are located data repositories and/or computer systems that can be used to run simulations.
- A potentially large number of *users* of NEESgrid equipment and resources, including earthquake engineers, students, and others.
- A variety of both campus and wide area *networks*, which provide (hopefully high performance) connectivity among the various elements just listed.
- An *operations center*, which provides various forms of monitoring and diagnostic facilities for NEESgrid as a whole.

In practice, individual sites may fulfill multiple functions. For example, equipment sites will include users and may offer remote access to data and compute resources.

During the three-year NEESgrid development and deployment process, we make a number of additional assumptions about NEESgrid structure:

- We assume that the operations center is located at the National Center for Supercomputing Applications (NCSA) in Urbana-Champaign.
- We assume initially a single central repository and simulation system, at NCSA, and extend the NEESgrid system over time to encompass other resource sites.

3.2 Capabilities

Based on the analysis in [4] (and the scenarios in Section 7) we identify the following capabilities that must be provided the NEESgrid software system. This list is not meant to be completely exhaustive (we provide more details in sections that follow) but does indicate the major concerns that we address in the design and helps set the scene for subsequent discussion.

Core capabilities. A small set of core capabilities cross cut essentially every aspect of NEESgrid operation.

- *Authentication and authorization*. Remote access to valuable equipment and other resources implies a need for robust *authentication* mechanisms (for verifying someone's identity) and *authorization* mechanisms (for determining who is allowed to do what). These mechanisms will be used to determine who is allowed to look at experimental results, configure experiments, run simulations, and so forth.
- *Resource discovery.* Remote access to equipment and other resources also implies a need for resource discovery mechanisms that can be used to determine the location of resources (e.g., available data repositories and compute resources) and their characteristics (e.g., number of video channels generated by an equipment site).

- *Resource monitoring*. Robust operation of NEESgrid implies a need for mechanisms that can be used to monitor the status of NEESgrid resources.

Data management. A second set of capabilities relate directly to the management of NEESgrid experiment data.

- *Metadata generation*. Tools are required for extracting or otherwise generating metadata, in some predefined format, for data generated by NEES experiments or simulations. (Note that while the definition of standard metadata formats is of great importance, we do *not* view the definition of these formats as part of the NEESgrid project.)
- *Data publication*. Tools are required for reliably and securely transferring data generated by experiments and simulations into data repositories subject to access control and indexing.
- *Data discovery*. Tools are required for discovering available data sets, based on criteria such as metadata.
- *Uniform resource access*. Tools are required for securely and reliably accessing remote datasets, ideally based on standard protocols and/or APIs so as to simplify the design of user data access tools.

Telepresence.

- *Teleobservation*. Tools are required for securely and reliably managing remote cameras and the video streams generated by those cameras, as well as for obtaining access to other relevant data streams from an equipment site.
- *Teleoperation*. Tools are required for securely and reliably managing remote experimental equipment.

Simulation.

- *Code repositories.* Tools are required for securely and reliably access repositories containing simulation codes.
- *Uniform resource access*. Tools are required for access remote compute resources, ideally based on standard protocols and/or APIs so as to simplify the design of user simulation tools.

3.3 NEESgrid Software Components

The NEESgrid system architecture defines a set of software services and packages that, collectively, address the concerns listed above. These software services and software can be characterized, first of all, in terms of where they are located.

- At the *operations center*, we operate the *NEESgrid Global Services*—a set of central services concerned with resource discovery, resource monitoring, and the like.
- At *equipment sites*, we place both standard hardware (the NEESgrid Point of Presence, or NEES-POP) and standard NEES-POP software that enables secure remote access to equipment for the purposes of teleobservation (and, ultimately, telecontrol), and secure publication of data produced by experiments to data indices and data repositories.
- At *resource sites*, we place software that enables secure remote access to resources for the purposes of data storage and access (in the case of data repositories) and simulation (in the case of compute resources);

- For *users*, we provide a small number of standard collaboratory access tools and a set of APIs and associated libraries that can be used to "NEESgrid-enable" other client tools.

In building the various services and packages that define the NEESgrid software, we use Grid services provided by the Globus Toolkit™ to address issues of authentication, authorization, resource discovery, monitoring, and uniform data access. Thus, for example, NEESgrid users need authenticate only once to obtain secure access to any NEESgrid resource that they are authorized to use.

4 NEESgrid Software Overview

We introduce here the software components that will be integrated to define the NEESgrid system architecture. This material provides context for the more detailed discussion in subsequent sections.

The NEESgrid software can be described in terms of the capabilities provided to users (e.g., a "data management service" for linking users with collaboratory data), in terms of the specific software to be installed at different locations (equipment sites, central operations site, etc.: see Figure 2), and in terms of *provenance* (which includes, as noted above, extant software from NMI and elsewhere, custom software developed by the NEESgrid team, and software provided by sites). All three perspectives are useful and are incorporated into our presentation in this section, which is structured as follows:

- We first describe the NSF Middleware Initiative (NMI) software base that we will provide a number of core NEESgrid services, focusing in particular on elements of the Globus Toolkit that will be used for authentication, authorization, resource access, resource discovery, and other purposes central to the goals of the NEESgrid collaboratory. (Other extant software to be used within the NEESgrid system will be identified in the sections that follow.)
- Then, we present the major global services that will be provided to address NEESgrid requirements for data access, authorization, and the like.
- Finally, we list the software that is to be installed at each of three different locations: storage resource, compute resource, and equipment site. In the latter case, most (but not all) of the required software is instantiated as part of the NEES-POP.

In each instance, we identify which elements of the architecture will be developed as part of the system integration activity, and also identify site-specific components that must be developed by participating NEESgrid equipment sites.

4.1 NMI Baseline

In developing the NEESgrid software, we will leverage software packaged under the recently funded NSF Middleware Initiative (NMI: www.nsf-middleware.org) project. This project funds the specification, packaging, testing, and support of a range of middleware services, of which the Globus Toolkit is a core element. NMI will produce a series of integrated and supported software releases over the next three years. The first release, NMI R1, is scheduled for spring 2002, consistent with the NEESgrid schedule of early adopter demonstrations in summer 2002 and release of NEESsoft-1 in fall 2002.

We note that Globus and hence NMI software releases will evolve over the three years that NEESgrid will be built, as a result of other funded R&D projects; standards activities in the Global Grid Forum (www.gridforum.org), Internet Engineering Task Force (www.ietf.org), and

elsewhere; and commercial adoption of, and support for, the technology. While the underlying conceptual framework of the NEESgrid architecture will remain intact, subsequent NEESsoft releases will benefit from (and, concomitantly, must be structured to enable) significant changes in the code base and the introduction of new services. We intend that the NEESgrid architecture follow this evolution at a pace that is consistent with its production goals. Thus, we expect to minimize risk by ensuring that NEESgrid is an instance of the more broadly deployed, commercially support production Grid infrastructure that we expect to see emerging in this timeframe.

4.1.1 Why NMI and the Globus Toolkit

Tracking the NMI release schedule has several key advantages from the perspective of the NEESgrid architectural definition. One of the goals of NMI is to have middleware infrastructure supported at the campus level. To ensure this deployment and acceptance, Internet-2 is one of the partner institutions that is participating in NMI. If NMI is successful on this front, NEESgrid will not rely on infrastructure that is deployed only for the purpose of earthquake engineering activities, but rather on supported campuswide infrastructure that is common with the broader scientific community on campus. This outcome is further reinforced by the fact that NMI is being targeted for the broad scientific community. For example, NMI software will be used to enable remote access to the TeraGrid (www.teragrid.org), which represents the largest numerical simulation resource in the United States for academic users.

A further advantage of basing the NEESgrid architecture on NMI is that we can focus NEESgrid *development* activities on software and services that are unique to the requirements of the earthquake engineering community, further increasing the likelihood of successful adoption by the earthquake engineering community.

The core of the NEESgrid system architecture will be the software and services provided by the Globus Toolkit, the de facto standard for Grid infrastructure that is being deployed by every major Grid project worldwide. (See www.mcs.anl.gov/~foster/grid-projects for a partial list of users.) The Globus Toolkit is an open source software base that provides the infrastructure services and the application programmer interfaces (APIs) necessary for constructing secure, robust Grid and collaboratory systems. An overview of the contents of the Globus Toolkit is provided in Section 4.1.2.

The Globus Toolkit has the backing of major hardware and software vendors such as IBM, Platform, Sun, and SGI. Part of this backing will be commercial support for Globus software, hence reducing the dependence of the NEES consortium with respect to long-term maintenance and support. Furthermore, major components of the Globus Toolkit are currently the subject of standardization activities in the Global Grid Forum (GGF) and the Internet Engineering Task Force (IETF). We also anticipate standards activities within the World Wide Web Consortium where appropriate.

The Globus Toolkit is distributed under a liberal open source license that allows the software to be used for commercial as well as academic development. Source code is available for all components and discussions of Globus development with the commercial setting have placed a strong emphasis on creating a single, common open source code base, much like that achieved for the Linux operating system and the Apache web server. The focus on open source and open standards combined with significant industrial involvement minimizes the risk associated with the use of Globus technology for NEESgrid.

4.1.2 Globus Toolkit Components

The Globus Toolkit defines protocols, interfaces (APIs), and services for basic collaboratory functions such as authentication (verifying identity), authorization (determining who can do what), resource discovery, resource access, resource monitoring, data movement, data replication, and so forth. These services are designed to be generic. That is, they are not targeted to any specific application but rather provide the basic building blocks from which many diverse applications can be constructed. The advantage of this approach is that the effort of constructing and maintaining this infrastructure can be shared with other disciplines, which is happening via, for example, the funding of the GRIDS Center. However, a consequence is that the Globus Toolkit mechanisms do not provide a complete NEESgrid software solution "out of the box": effort must be devoted to construct an application via the integration of appropriate Globus services.

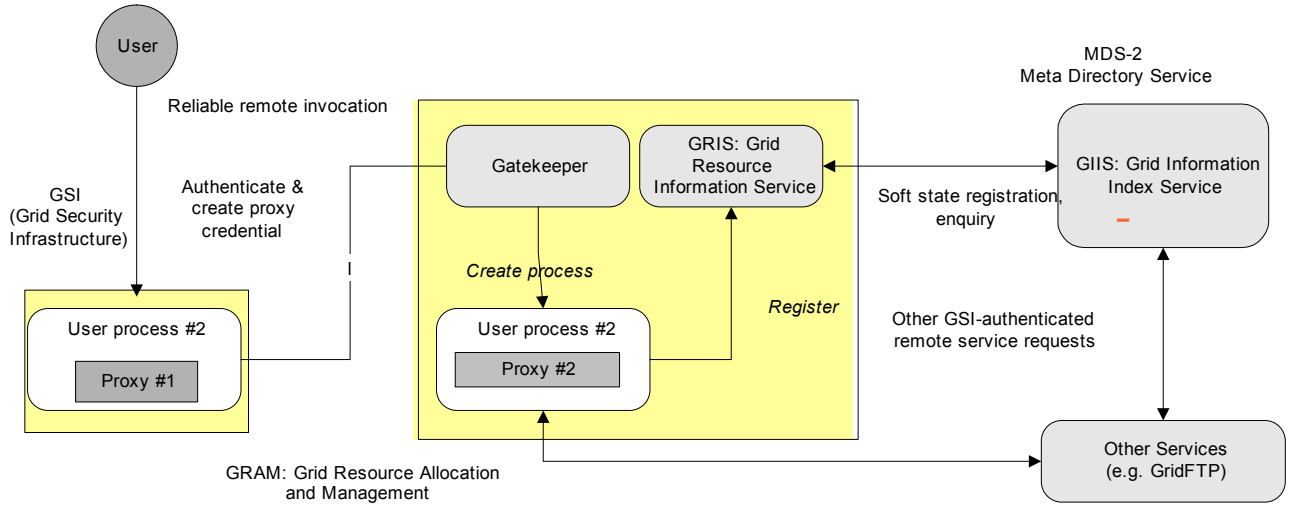

Figure 3: Overview of primary Globus services and protocols.

Figure 3 summarizes the basic services and protocols provided by the Globus Toolkit. We provide a brief summary of these services here. More detailed discussion is provided in later sections.

The Grid Security Interface (GSI) [6] addresses authentication and authorization issues. GSI defines all security protocols as well as defining a standards-based security API (the Generic Security Services API [7]). All interactions with Globus services are mutually authenticated, meaning that both the requestor and requestee must prove their identity. Globus authentication mechanisms are based on public key authentication algorithms and on widely used and deployed standards such as the X.509 standard for public key credentials [3]. We provide a more detailed discussion of how certificates will be managed within NEESgrid in Section 5.1.

A significant element of the Globus Toolkit security model is the use of delegation to provide a *single sign on capability*. Single sign on is important for NEESgrid because it enables a user to authenticate once to the NEESgrid environment and then have access to all NEESgrid resources (subject, of course, to access control and policy restrictions) without having to "log in" to each individual resource individually. This is important, not only for convenience, but also for telepresence where distributed resources such as telecontrol interfaces, simulation and online

storage must be coupled within a single experiment. Delegation is implemented by the creation of *proxy credentials*, which enable the possessor to act on behalf of the user that created the proxy.

Figure 3 illustrates the two primary locations where proxies are created and used within the Globus Toolkit. First, a user creates a proxy to "log in" to NEESgrid. This approach increases the overall security of the NEESgrid architecture by limiting the use of a user's actual credentials to the creation of the proxy. Second, proxies are created whenever processes are created on behalf of a user via Globus Toolkit resource management operations. This proxy allows the created process to interact with other system components on behalf of the user. For example, in the use case scenario described in Section 7, a running simulation computes the parameters for the next actuator setting in a pseudo-dynamic test. In this situation, the simulation needs to convey these parameters to the telepresence interface. Because the simulation possesses a proxy that indicates that it has been delegated rights from a specific user, the simulation can authenticate on behalf of that user to the telepresence interface and perform the required control operations safely.

Globus Toolkit resource management services are currently focused on the allocation and management of computational resources. The Grid Resource Allocation Management (GRAM) protocol defines a network protocol for remotely initiating resources on a wide variety of computational resources including clusters and shared memory parallel computers. Resource management services are implemented by a secure service called a *gatekeeper,* whose role is to start processes on a resource on behalf of the user. The gatekeeper uses GSI authentication and runs on a standard port registered with the Internet Assigned Names Authority (IANA). The Globus toolkit include C and Java client APIs along with a variety of command line tools for submitting and managing computational tasks.

Another critical Globus Toolkit service is the Meta Directory Service (MDS), which provides a means for discovering Grid resources along with their characteristics (e.g., the number and type of video feeds from a remote experiment, the bandwidth of a network, the total capacity of a storage system). MDS provides standard interfaces for *service data* about a single instance of a service (provided by that service itself), as well as an *index service*, which collects and reports on service data collected from many services, as well as information not directly related to a service (such as a host's CPU type and operating system). MDS also uses GSI-based authentication. One important role of the index service is to support the creation of *virtual organizations* by enabling members of collaboration to discover the properties of resources that are available to that collaboration. Thus, for example, NEESgrid will operate a set of index servers to maintain information about available equipment and data resources.

The Globus Toolkit currently provides two services for managing large distributed data sets:

- GridFTP, an extension of the FTP protocol that incorporates GSI authentication, parallel third-party and partial data transfers, for which both clients and servers (as well as libraries supporting lower-level APIs for application programmers) are available.
- Replica Location Service (RLS), a distributed discovery service for locating replicated data files.

GridFTP and RLS clearly represent an important but limited subset of the services that are required for the full range of Grid-based data management tasks associated with NEES. In particular, the Globus Toolkit does not currently support interfaces to databases or provide services for managing metadata. We propose the use of non-Globus Toolkit protocols (see Section 5.2) to address these limitations. We expect that future versions of the Globus Toolkit will directly support metadata management as a result, in particular, of major development efforts under way in the United Kingdom as part of their eScience program.

4.2 Centralized and Global Services

NEESgrid provides the following systemwide services to support the specific mission of the NEES program:

- *Data services*, including a central repository for storing NEES data, a location service for replicated data, and a metadata management system for attribute-based querying of NEES data.
- *Information services*, for discovering and monitoring of NEESgrid resources.
- *Community authorization service*, for managing community membership and authorization policies governing access to NEESgrid resources.
- *Software repository*, for providing access to simulation codes.
- *Collaboration services*, for facilitating community-wide interactions.

In the initial phases of NEES, the central components of these services will run within the NCSA production environment. In addition, secondary backup versions of these services will be made available in some instances (for example, for replica location and top-level information service). A brief description of the central services follows. More detailed discussion of individual services appears later in this document.

4.2.1 Data Services

Centralized data services provide the various capabilities required to support the community-wide NEESgrid data repository. Note that from the perspective of implementation, interfaces, and services, the centralized repository is identical to all other NEES data repositories. Thus, all data services associated with the centralized NEESgrid data repository are also associated with any other NEESgrid data repository. Centralized NEESgrid data services are as follows:

- *Data access/movement service*. This service provides high-performance access to data stored in the repository via the GridFTP protocol. The GridFTP server to be used will support single sign-on with NEESgrid credentials, community-based access control, third-party (server to server) data transfers. The use of GridFTP will make NEESgrid repositories interoperate with other Grid data repositories. Interoperability with non-GridFTP repositories is via a data gateway service that provides a GridFTP protocol interface to legacy data repositories.
- *Replica location services*. As the number and size of community data sets grow, it can be desirable to replicate certain data sets in order to ensure increased reliability or higher-performance access. While NEES data will initially be small in comparison with that managed by other communities such as high energy physics, we anticipate that the amount of data in the NEES repository will grow rapidly to the point where replication is desirable. For this reason, the NEESgrid architecture supports a replica location service [5], which enables a user or application to discover where copies of a desired data file are physically located. This service has two components: a local catalog, in which the contents of a specific repository are made known, and a global index service, which keeps track of which storage systems contain which files. The central NEESgrid repository will maintain a catalog of files stored within it. NEESgrid global services will include a top-level index of all NEESgrid repositories for the purpose of replica location.
- *Metadata management services*. The NEESgrid system architecture supports *attribute-based data discovery*. That is, data elements can be identified not just by file name, but

also by attributes of the data itself, such as the name of the sensor that generated the data, the date of experiment, or the name of the experiment. Attribute-based discovery requires the ability to *harvest* metadata, using it to annotate existing data files, and the ability to query the metadata to identify a data file that contains the needed data. A harvesting and discovery service will be supported by the central NEESgrid data repository. More details on these services are provided in Section 5.2.

4.2.2 Global Information Service

The NEESgrid system comprises many separate, individually operated components, including computers, data repositories, and experimental facilities. The distributed and dynamic nature of these components makes knowing what resources are available to the NEES community a challenging problem. This problem must be addressed if the community is to make effective use of NEESgrid. Thus, the NEESgrid architecture provides a top-level index server that serves as a single point of contact for information about NEESgrid services and resources. All NEESgrid resources are required to register with this index using the registration protocols defined by the Globus Toolkit (these registration and query protocols are collectively referred to as MDS). From this index, one can navigate or search to locate other NEES data repositories, compute resources, and the like, regardless of what organization maintains the underlying resource. Information services are discussed in more detail in Section 5.3

4.2.3 Community Authorization Service

Access to NEESgrid resources will be controlled on a per user, per group, or community-wide basis. For example, permission to observe the results of an experiment may be granted to the NEES community as a whole, while permission to add annotations may be restricted to engineers collaborating on that experiment, and permission to control the experiment is limited to the experiment's principal investigator. Within the Globus Toolkit, the Community Authorization Service (CAS) [8], discussed in detail in Section 5.1.3, provides mechanisms for defining groups within a community and assigning rights to community members. The NEESgrid system architecture specifies the operation of a single NEES-wide authorization service.

4.2.4 Software Repository

The NEESgrid system architecture defines a standard secure interface, GridCVS (www.globus.org/gridcvs), to CVS code repositories used to maintain community simulation codes. The NEESgrid system operated by the NEESgrid team will include a single central code repository located at NCSA; other repositories may be operated by other NEES participants.

4.2.5 Collaboration Services

NEESGrid will have a central web server with a standardized set of collaboration tools enable users to work collaboratively across sites. The Collaboration software is based on a Grid-enabled version of the CHEF (www.chefproject.org) collaboration toolkit.

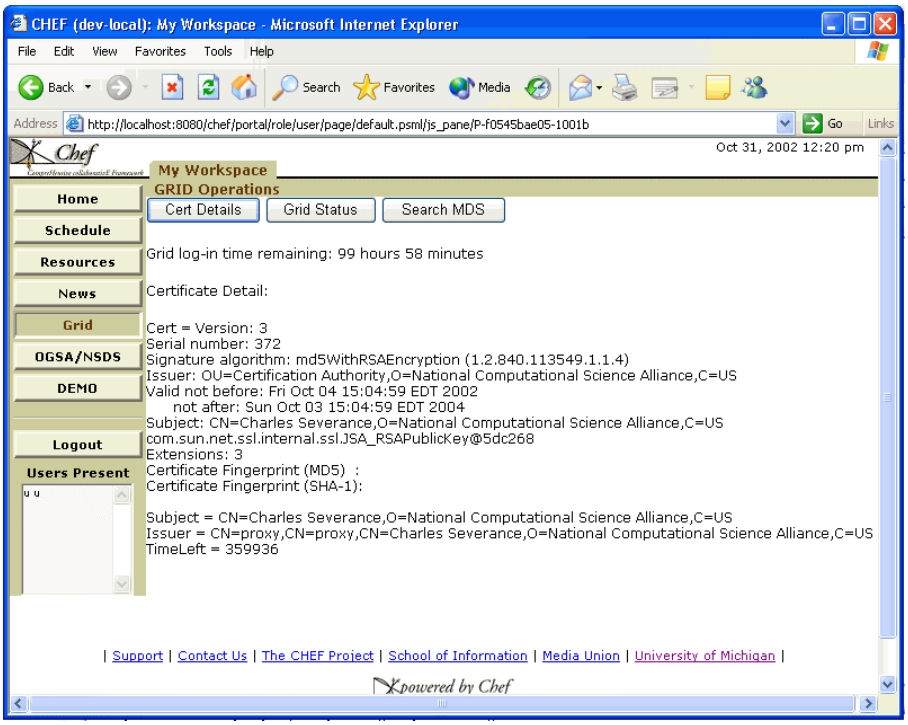

Figure 4: CHEF user interface

This software includes the following capabilities:

- Document Sharing
- Calendaring and Scheduling
- Mailing list support
- Chat
- Threaded discussion
- A portal interface to grid tools – Certificate management and web-based Grid FTP
- Streaming data viewer capable of live monitoring of properly instrumented simulations and experiments.
- Electronic Notebook

The security and authentication for the web-based collaboration tools is based on the myproxy capability of the Grid. Users "check-in" their credentials using myproxy and the web server checks out the credentials when the user logs in.

The collaboration toolkit is useful for large and small groups. A group may be as large as the entire NEESGrid community or as small as a researcher and a few graduate students. Each of the (overlapping) groups will have access to their own instances of each of the tools.

4.2.6 NEES-Specific Components of Central Services

Central NEESgrid services are, for the most part, configurations and deployments of standard NMI software components. Nevertheless, limited elements of the NEESgrid central services will have to be developed specifically for this application. These are as follows:

- Metadata harvesting tools that can accommodate the variety of metadata generated and used by NEES sites and users. The NEES Metadata Harvesting Protocol will be developed as an extension to existing NMI services, to provide a means of providing the central repository with access to metadata from sites. In addition, the SI will support community development of NEES-specific models and formats. This activity will as much as possible leverage existing metadata practice by the sites and within the community
- Data discovery tools that support attribute-based location of data sets in NEESgrid repositories. Again, these will be implemented as extensions to existing NMI services. These will enable NEES users during, for instance, collaborative sessions, to discover the location of data and metadata items of interest in NEESgrid repositories, which can then be accessed using NMI services such as GridFTP.
- Administrative tools for managing data repositories and access to data. NEESgrid will provide basic tools for managing repositories, including facilities for adding, deleting, and moving datasets from repositories; adding, deleting, and editing links between data and metadata items; and editing access control parameters for data items.

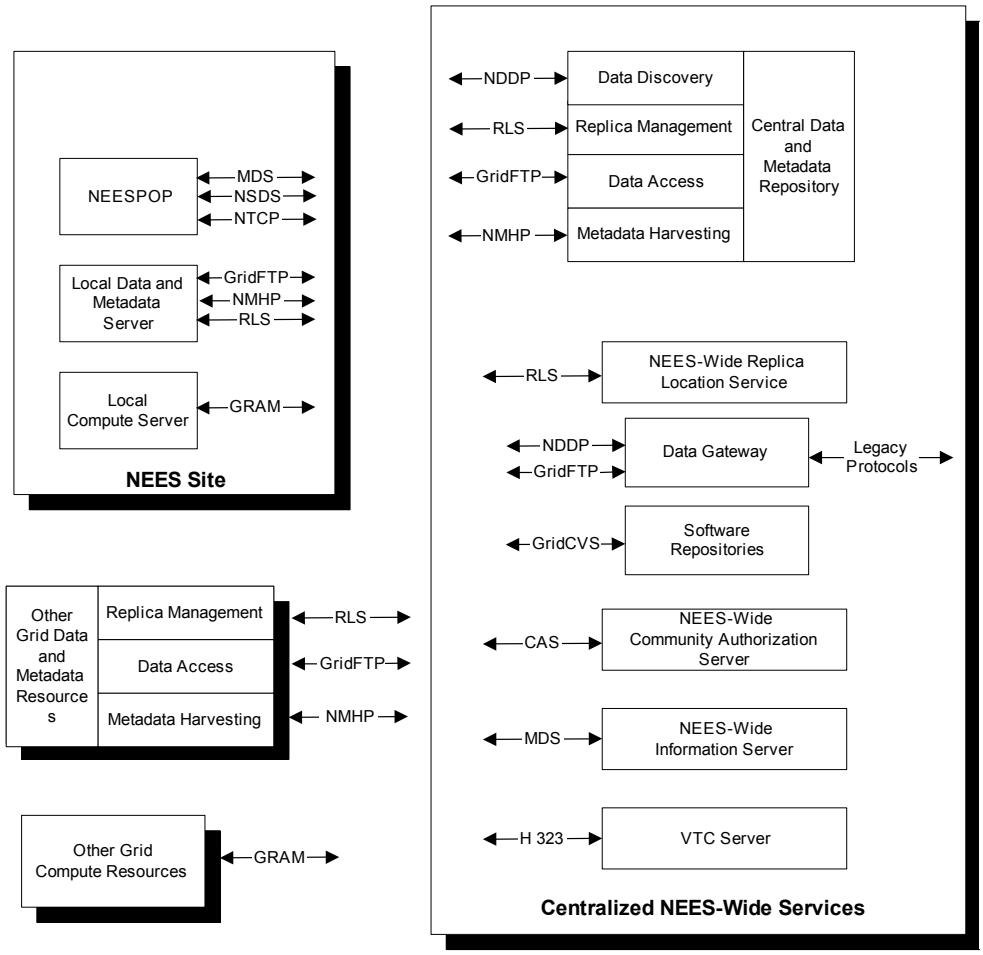

Figure 5: Global structure of the NEESgrid system architecture

4.3 Grid-Enabled Storage Resources

The NEESgrid system architecture supports the integration of data repositories outside of the central repository. These may be associated with an experimental site, or they may be stand-alone data caches. Full-fledged NEESgrid repositories require that harvesting and discovery protocols be supported as well. We don't anticipate that general Grid storage repositories will conform to this requirement. Therefore, it is expected that NEESgrid will be able to move data to and from any Grid enabled storage system, but harvesting and discovery services will have to be delegated to NEESgrid-compliant storage sites.

4.4 Grid Enabled Compute Resources

The NEESgrid system architecture allows for the integration of other Grid-enabled compute resources such as TeraGrid compute engines and machines operated by other research groups or computer centers. Any compute platform that is running NMI software, specifically the GRAM resource management protocol and the GRIS information service, can participate as a full-fledged member of NEESgrid.

4.5 Equipment Site System Architecture

The specialized and heterogeneous nature of the equipment located at different NEES equipment sites complicates the task of providing standard interfaces to that equipment. For this reason, the NEESgrid system architecture introduces the notion of the NEES-POP, hardware deployed at all NEESgrid equipment sites to run key services. Thus, the NEESgrid system architecture defines an equipment site in terms of not only the standard services that must be supported (as is the case at resource sites, etc.) but also a standard NEES-POP hardware configuration. Any equipment site that conforms to these architectural specifications can participate in the NEESgrid system.

A NEES equipment site must provide both a NEES-POP and a "local" storage repository; it may also provide compute resources and additional storage repositories. Figure 6 shows the major architectural components deployed at a NEES equipment site. The shared box to the right indicates the components that are deployed as part of a NEES-POP (described in Section 4.5.7).

The services provided at an equipment site are as follows:

- *Resource-specific information services* for characterizing the experimental, data and compute services offered by the facility,
- *Telepresence services* for remote observation and control of experimental equipment,
- *Site data services* for storage and immediate access to experimental data,
- *Collaboration services* to support multisite experimentation, and
- *Optional compute services,* to facilitate simulation and data analysis.

Increasingly sophisticated instantiations of each service will be provided in each NEESgrid software release, from NEESsoft-1 onward. We expect the most evolution in the telepresence services, as most of these will be constructed specifically for NEESgrid. We now describe each service briefly, with more detailed discussion provided in Section 5.

4.5.1 Resource-Specific Information Services

Equipment sites provide a diverse set complex services. Resource-specific information services provide a standard means of querying the experimental site to discover what types of services are offered and how they are configured. For example, the configuration of the teleobservation service (e.g., the available control and observation points and what are they observing) would be provided. The configuration of an optional compute resource is another example of the type of information that is provided via the information service.

Information services are provided by two kinds of server: a resource server (such as a telecontrol server) built on the Globus Toolkit may report service data (such as which control points are available), using a standard interfaces for queries and subscription, and an index server collects information from various sources and makes that information available using standard query and subscription interfaces. Index servers collect information both from resource servers and from information providers, programs which collect data not associated with a service. Some information providers are supplied with the Globus Toolkit (including providers that report information about hosts, such as the CPU type, operating system, and current system usage statistics); sites may write and deploy additional information providers (for example, to report information about a non-Globus based service).

Globus-based services being developed for NEESgrid will report meaningful service data; additional information providers will be developed as appropriate. The central NEESgrid

information service identified in Section 4.2.2 will be an index server that aggregates information from all NEESgrid sites.

4.5.2 Telepresence Services

Telepresence services distinguish a NEESgrid equipment site from any other NEES resource sites—and, indeed, from most other Grid sites. The telepresence interface provides the following capabilities:

- Direct the flow of experimental data from local data acquisition hardware to a storage resource so that it (and associated metadata) may be accessed via the site data services (described in Section 4.5.3).
- Facilitate direct observation of subsets of sensor data as they are produced, subject to policy and resource availability (network bandwidth for example).
- Enable setting of experimental parameters, including initiation of actuators. The scope of this operation is subject to local policy, safety issues, and experiment policy.

Detailed data flow for the telepresence service is presented in Section 5.4. Note that the telepresence service depends on the existence of a local data repository. This service must be provided in addition to the NEES-POP.

4.5.3 Site Data Services

The primary function of equipment site data services is to provide a standard Grid-enabled location on which data from experimental runs can be stored. Site data services are identical in their structure to the central data services, consisting of a data access service, a metadata harvesting service, and a data discovery interface.

The existence of site data services means that a remote user of the experimental facility does not have to understand the details of the local data acquisition hardware. It also allows the experimental facility to isolate the data acquisition hardware and any associated storage from the open network. All data and metadata stored on the side data service must conform to NEES data standards.

We note that site data services are operated under policy determined by the site. In particular, the site does not have to offer long-term archival storage of experimental data. Rather, it may require experimenters to transfer data to other specialized repositories or to the NEES central repository.

4.5.4 Collaboration Services

Each site will have their own web server with the collaborative tool kit installed and configured on their NEESPop. This allows each site to create their own groups of collaborators. Since the entire NEESGrid used Grid credentials for authentication and identification, users will have the same account and password regardless of which collaborative environment they are using. In addition since all of the sites will use the same software with the same look and feel users will be able to move easily between the various NEESGrid sites.

4.5.5 Optional Compute Services

Equipment sites may wish to integrate local compute resources into NEESgrid, for purposes of simulation and data analysis. Such local resources will be particularly important when numerical simulation is an integral part of a test and the long and sometimes indeterminate latency of wide area network connections may make the use of remote resources infeasible.

In spite of their close proximity to the experimental apparatus (indeed, because of that proximity), it must be possible to manage site compute resources remotely. Thus, these resources should be *Grid enabled*, by which we mean that they should be set up to respond to the GRAM resource management protocol and to provide configuration and status information via Globus information protocols. Thus, site compute resources are indistinguishable in terms of function and services from any other Grid-enabled compute resource within NEESgrid.

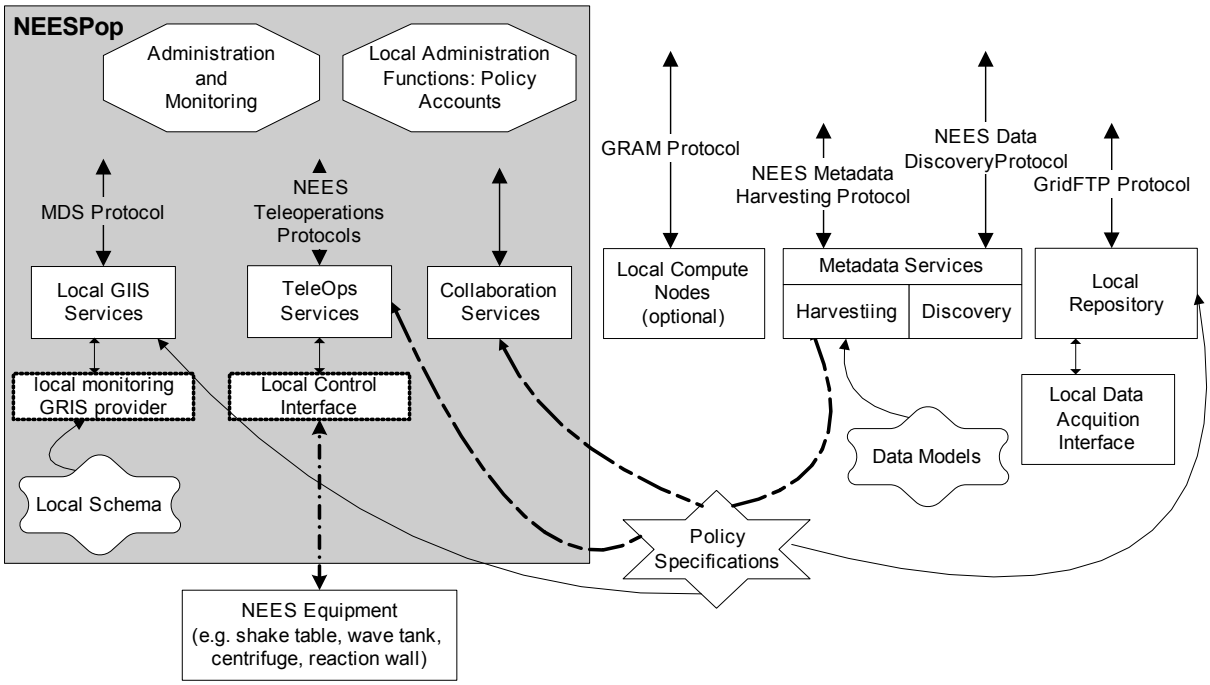

Figure 6: Details of NEESgrid architecture from the perspective of an equipment site.

4.5.6 NEES-Specific Equipment Site Components

NEES-specific equipment site components include the following:

- Interface software to link telepresence services to local data acquisition systems.
- Site-specific schema and information providers, especially for experimental apparatus.
- Experimental metadata capture and data reformatting tools to interface into local repository and metadata harvesting and data discovery services.
- NEES data models (same as required for global repository).

In many instances, implementations of the NEES-specific components can be shared across equipment sites. In some situations, however (for example, if custom data acquisition software is used), these components may have to be modified or even written specifically for a particular equipment site (this is discussed more fully in Section 5.4).

4.5.7 The NEES-POP Concept: Mediating NEES Equipment Site Access

A NEES equipment site operates one or more pieces of specialized equipment, such as a shake table, tsunami wave tank, or centrifuge. A central goal of NEESgrid is to allow sites to make these resources accessible to remote NEESgrid users, subject to local policy constraints. A site may also have other resources that it wishes to make accessible to the community, such as computers or data archives, and we wish to enable remote access to those resources also.

The specialized nature of earthquake engineering equipment means that different sites tend to have unique characteristics in a wide variety of areas, including access policies (who can and cannot access the resource), remote control/steering interfaces, performance enhancements such as data caches, and publication of information such as health/status, reservations and more. While these unique characteristics are manageable at the local site, they represent a significant obstacle for remote users, who cannot afford to learn different access mechanisms for each remote site. Just as we use Web browsers, secure shell (ssh), and ftp to access remote data and computing, we need standard mechanisms for accessing remote equipment resources.

Standard access mechanisms are needed for a number of purposes, including the following:

- *NEESgrid system managers* want to be able to monitor network connectivity to sites and diagnose network accessibility and performance problems.
- *Remote users* want to be able to determine the status of equipment, to determine for example the schedule of experiments and current equipment state. They also want to be able to access teleobservation feeds, talk to operators, access experimental data, and so forth.
- *Equipment site operators* may want to be able to structure interactions among equipment at two or more equipment sites, in the case of multisite experiments.

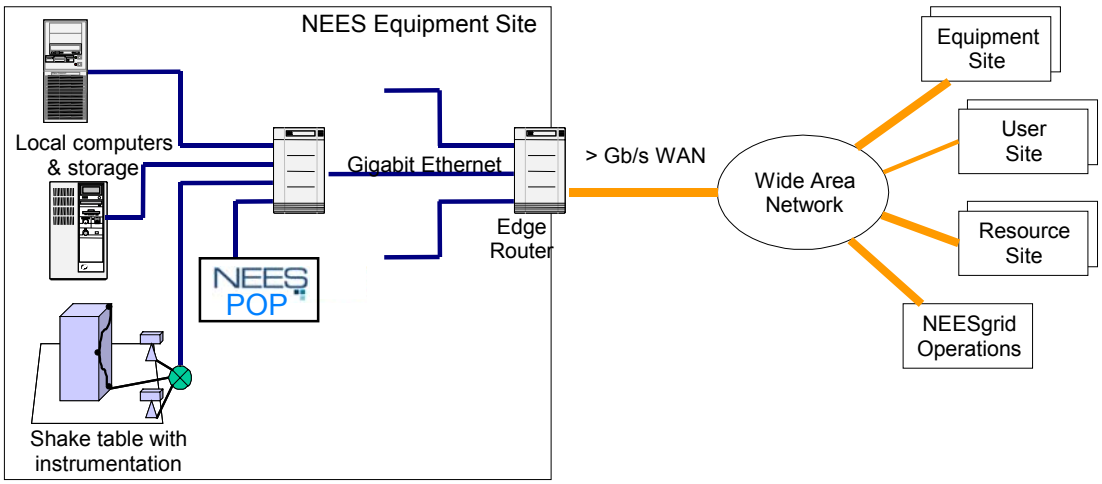

Figure 7: NEESgrid schematic, showing an equipment site on the left, with its NEES-POP, and wide area links to other NEES resource and user sites.

The overarching requirement is for mechanisms that mediate between a site and the outside world. As illustrated in Figure 1, the NEES-POP provides a standard, conveniently located environment in which these mechanisms can operate. The services supported by the NEES-POP are specified in Figure 7.

From an equipment site perspective, the following aspects of this system are important:

- Cost is low, estimated at around $5,000.

- The base software configuration is standard Linux, with additional standard and exhaustively tested software provided by the NEESgrid team. Remote administration by the NEESgrid team is provided, so local administration costs associated with the NEES-POP should be minimal.
- Extended software services can be provided to mediate access to specialized devices—providing, for example, online access to data from a non-Internet Protocol-addressable device. The nature of these extended services need to be defined in consultation with equipment sites and the NEES user community. NEESgrid personnel plan to assist with the development of such software by providing toolkits, but ultimately the community will have to negotiate appropriate tradeoffs between development effort and ease of remote access.

The advantages of defining and deploying this standard component include the following:

- We reduce the number and complexity of the human interactions required to add a new equipment site to NEESgrid. For the NEES-POP, we simply point a site at a web page listing the required equipment. (In fact, we can probably also point them at a vendor who can deliver the package.) This simplicity is important when we are talking about tens of sites: even just two weeks of one-on-one effort per site adds up to person-years.
- The detection and correction of the operational problems that inevitably will arise are simplified.
- The NEES-POP equipment provides a neutral zone for installation of monitoring equipment and software that the NEESgrid team (and subsequently NEES Consortium) can use to detect problems and optimize NEESgrid operation.

We emphasize that deployment of NEES-POP equipment and software upgrades is *desirable*, because it enhances remote accessibility. However, NEES-POP equipment and software is never necessary for local operation, and thus its availability and correct operation will never be on the critical path for deployment and upgrading of local equipment.

4.5.7.1 NEES-POP Overview

Essentially the NEES-POP is a PC running Linux and standard NEESgrid "middleware" services that link site resources into the national-scale NEESgrid collaboratory. These services may include the following:

- *Information services* allow remote users to determine the characteristics and status of site resources, including availability, software versions, and tool location, with access control if desired. These services can also provide access to performance information on networks and local resources, obtained via passive performance sensors and/or active network performance tests
- *Data management and caching services* use NEES-POP storage as an intermediate staging area for data being made accessible to remote users.
- *Equipment management services* map from appropriate instrument monitoring and control services to the mechanisms used to monitor and control specific research equipment.
- *Collaboration services* provide local support for services to support distributed scientific team-based work, via such mechanisms as document/file repositories, discussion groups, email archives, announcements, chats, calendars, and conferencing. In practice, this is

likely to mean a local Apache service with Tomcat, so that collaboration capabilities can be downloaded as required.

- *Security services* support NEESgrid-wide identity and certification procedures used to verify the identity of requestors, and the associated mapping from global to local identities and local authorization [2]. While site administrators need not require that all remote accesses pass via their NEES-POP, they can do so if they wish, obtaining increased security.

In the following, we provide more details on hardware and site configuration and administration issues.

4.5.7.2 Hardware Configuration

Our minimum recommended hardware specifications are as follows. This specification assumes that the NEES-POP is a dedicated device and is not being used for other tasks. The 36 GB disk space provides minimal long-term data storage. If the site plans to use the NEES-POP to store instrument data, additional disk will be required and will need to be sized based on application requirements. Two network interfaces are specified so that both an internal LAN and a separate external WAN network can be supported.

- 1 GHz or greater dual Pentium-III or Pentium-IV processor
- 1 GB RAM
- 2 x 64-bit, 66 MHz PCI slots
- 36 GB SCSI disk
- 2 x Gigabit Ethernet NICs

At the time of writing, a system with these characteristics can purchased for around $5,000. NEESgrid personnel will develop standard language that can be used when writing bid specifications for this equipment.

The NEES-POP may also be provided with a Global Positioning System (GPS) receiver, allowing for accurate time synchronization.

4.5.7.3 Placement and Administration

As specified in the NEES solicitation, NEES sites are expected to have at least Gb/s internal network connectivity. Ideally the NEES-POP itself should have a direct connection to the WAN network; however, that is considered unlikely for most cases. Thus, a GigE network path will be required all the way to the campus edge router, in order to permit meaningful network performance experiments. It should *not* be on a normal campus backbone where it is competing with other substantial traffic, such as Napster MP3 downloads. The internal network between the NEES-POP and the instrument should be a standard Gigabit Ethernet LAN.

The NEES-POP will be administered by both the local site and the NEESgrid integration team. In practice this means that both local site administrators and approved NEESgrid integration team members will have root privileges on the machine and can install and reconfigure software under agreed-upon rules of operation.

NEES-POPs will be remotely monitored by the operations staff at NCSA for reliability.

A NEES-POP Working Group will be established, comprising developers, site administrators, and NEESgrid users willing to assist in NEES-POP specification and testing.

4.5.8 Security Considerations

It is critical that the NEES-POP not weaken a site's security. Linux systems, like other commercial operating systems, are a frequent target for hackers, and out-of-the-box configurations are an easy target. NCSA has expertise in protecting such systems; indeed NCSA's supercomputer consists of hundreds of systems running Linux. NCSA will leverage its experience to maintain the highest level of security on the NEES-POPs. For the most part this translates into constant attention to configuration details, reported bugs, and establishment of procedures. The systems will be configured with only the necessary services, thus reducing potential security vulnerabilities. NCSA belongs to a number of security watchdog communities that give advance warning of impending attacks. We will use this advance notice to (1) notify the NEESgrid Security team of impending danger, (2) develop a defense plan, and (3) test, document, and deployed solution (patch) in a timely fashion, when required, to all NEES-POPs through an automated mechanism run centrally from NCSA. Further NEES-POP events will be closely monitored at NCSA for any sign of an attack.

NEES-POPs can also be used to provide enhanced security-monitoring capabilities for other NEESgrid resources. Security events such as the information normally published to a system logfile can be pushed to the NEES-POP loghost. This information can then be monitored by automated systems at NCSA to scan for a set of predefined events that trigger a response. The information published to the loghost will need to be standardized through the NEESgrid Security Working Group, and each site will need to document appropriate response procedures. Responses could range from an email to a local site administrator to shutting down the entire site. NCSA will monitor security events 24/7, but local sites will determine the appropriate reaction. This strategy will not replace site security mechanisms but can serve as an important augmentation.

5 Details of NEESgrid Components

5.1 Security Components

Each of the above scenarios involves a security requirement—only authorized individuals should be able to read or modify experiment data, run simulations, and so forth. NEESgrid architecture must address these areas related to security: the overall security policy, authentication, integrity and confidentiality of data over the network, and authorization. And, although operating-system-level security is beyond the scope of the NEESgrid project, the NEESgrid architecture must not impose any requirements that would compromise site security.

5.1.1 Overall Security Policy

The NEES community as a whole, as well as specific sub-communities within NEES, will need to develop an overall security policy, covering such issues as defining the membership of the NEES community, defining acceptable use policies for NEES resources, and determining who will run essential NEES-wide security services. As part of our requirements-gathering activity, we will help the NEES community define this policy. As with Grid services in general, we expect security requirements to evolve over time and will adapt the NEESgrid architecture to accommodate this evolution.

5.1.2 Authentication, Confidentiality, and Data Integrity

We plan to base NEESgrid security on the Grid Security Infrastructure (GSI) introduced in Section 4.1. This is a state-of-the-art public key technology-based authentication system that has been widely used in Grid environments. GSI provides single sign-on capability with the ability to

provide secure and trusted authentication of every user and resource, as well as mechanisms to ensure confidentiality and integrity of data in transit over the network.

Public Key Infrastructure (PKI) is based on third-party trust. A third party essentially vouches for the identity of an individual or other entity. The physical "proof" then is held within a digital certificate that has been digitally signed by this third party. In this way two parties that otherwise do not have any relationship can trust each other's identity. A typical use of this certificate is to authenticate to a compute resource. Hence, instead of presenting a password, the party (or program) presents this digital certificate as "proof" of identity.

To establish trust, each party must have faith in the third party that certifies identities. This third party is a Certificate Authority (CA). Trust of the CA is established through a Certificate Policy. This policy is written typically by the relying parties, which are those that are taking the risk accepting identities. This Certificate Policy (CP) states all policies related to how the CA establishes identity and how they manage keys and certificates.

PKI is a standard Grid authentication technology and meets one of our goals to allow for the interoperability beyond NEESgrid. Building a PKI can represent a significant effort, so we plan to use the Public Key Infrastructure for creating credentials and distributing them as described below in Section 5.6.1.

The Alliance CA enforces a certificate policy that likely already meets NEESgrid requirements. As part of this effort, however, we will work with equipment site representatives to define a NEESgrid policy. Since the Alliance CA was developed, an effort has been made to standardize certificate policies for Grids, educational institutions, and government agencies.

5.1.3 Authorization

Support for authorization in NEESgrid will be provided via two complementary mechanisms: the *identity mapping* feature provided by GSI and a *Community Authorization Service* that extends GSI.

Traditionally, authorization in GSI has been accomplished through identity mapping: site administrators at each site create local login accounts for each authorized grid user and maintain a database that associates GSI (X.509) subject names with those local user names. The site administrators (and other local users) then use local access control mechanisms to grant access to those local login accounts. When a GSI-enabled server that uses identity mapping for authorization receives a request, that server uses the identity mapping database to find the local user name associated with the subject name from the user's credential and gives the remote user the same permissions as the corresponding local user (this is often accomplished by using an operating system call to run as that local user).

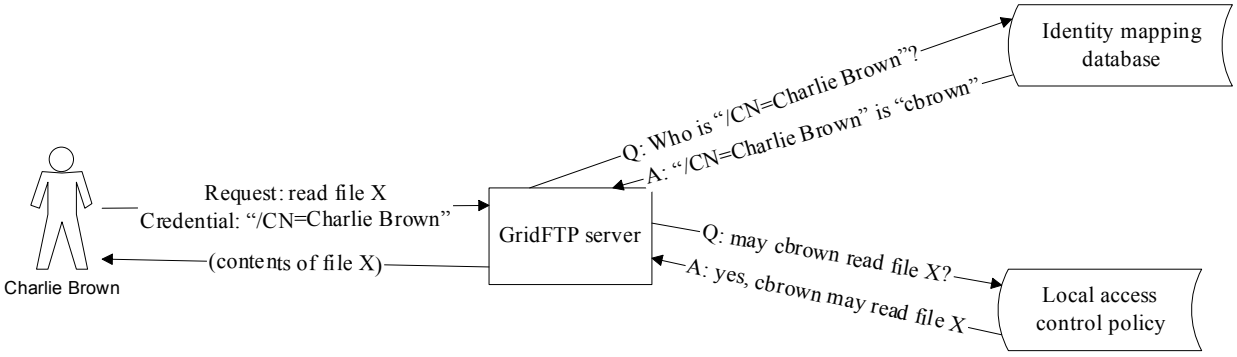

Figure 8: Identity mapping in GridFTP

In many cases, the mechanisms for access control via identity mapping are straightforward and easy for site administrators and users to understand. Site administrators use the same familiar mechanisms to create local accounts for remote users that they use to create accounts for local users; users and administrators grant access to remote users with the same familiar mechanisms (configuration files and OS-level commands to change file permissions, set quotas, etc.) that they use to grant permissions to local users.

In some cases, however, the administrative burden associated with identity mapping becomes significant. When a new user enters a virtual organization (e.g., when a new graduate student begins work on a project associated with NEESgrid), administrators at each site that the new user needs access to must create a local account for that user. When a researcher wants to grant access to a set of resources at several sites, that researcher must perform the appropriate access control commands at each site; this may be a complicated task, as the researcher must now know how to use the local access mechanisms used each of those sites, and must also know the user's local account names at each site. To alleviate this burden, we provide a Community Authorization Service (CAS) to support the distributed administration of community-wide authorization policies, while providing for overall local control of resources.

In the CAS model, site administrators use the identity mapping mechanism described above to grant access to a community as a whole to some set of resources; for example, the administrator of a file server might grant access to one disk to the NEES community. Representatives of the community (for example, members of the NEES Consortium) then run a CAS server, which maintains a database of fine-grained community policies (e.g., who exactly is allowed to read or write which files). A user who wants to access a resource will first connect to the CAS server to acquire a credential that includes embedded policy information, and will then use that credential to authenticate to a resource server. For example, a user might receive a credential from a CAS server with embedded policy that states that the user may read certain files, and then use that credential to authenticate to a GridFTP server. When the resource server receives the request, it verifies both that the local policies authorize the request for the issuer of the credential (e.g., that the requested file lives on the disk that was granted to the NEES community) and that the credential's embedded policy authorizes the request. Thus, the user effectively receives the intersection of the set of rights that the local administrator has granted to the community and the set of rights that the community has granted to the user.

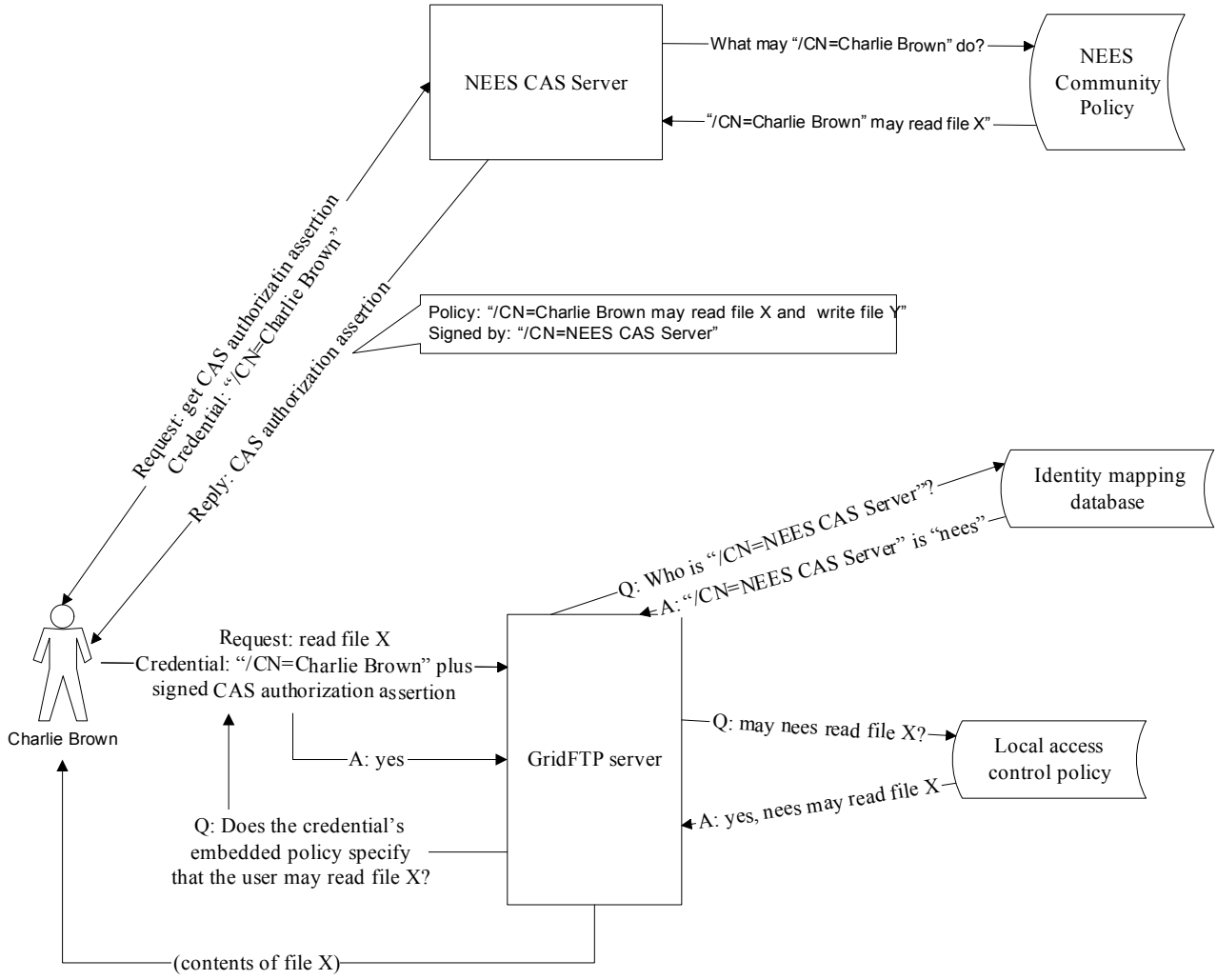

Figure 9: Using CAS with GridFTP

The CAS interface allows for the granting of rights to perform actions on individual resources or groups of resources to individuals or to groups of people. The CAS server itself will be administered centrally; however, the administration of authorization policies (i.e., what users and what objects are in what groups and what users or groups of users are granted which rights on what objects or groups of objects) will be distributed. The CAS server implementation supports a broad spectrum of access control models; depending on the policies chosen by the NEES community and the NEES Consortium, the ability to create and maintain groups of users and resources, to add new users to the community, and to grant access to groups of resources to groups of people may be centralized or delegated out to members of the community.

For example, the authority to create groups may be kept central, the authority to add members to the *tsunami-researchers* group may be delegated to a set of researchers within the community, and the authority to add members to the *shake-table-operators* group may be delegated to a different set of people within the community.

The initial CAS server, as well as client interfaces to perform policy administration (manage groups, grant and revoke access, etc.), has been developed. Use of the CAS for authorization to a particular service requires that the server providing that service be modified to recognize the credentials issued by the CAS server. A CAS-enabled GridFTP server has already been developed; CAS-enabled servers for other NEESgrid services will be developed incrementally over the course of the project.

5.1.4 Site Security

Some equipment sites may choose to isolate their data acquisition and control systems on a local area network that is not accessible to the outside world. Although the NEESgrid architecture does not require or implement this topology, it is compatible with it. The NEESgrid architecture does not require that the data acquisition and control systems communicate with any host other than the NEES-POP.

5.2 Data Components

NEESgrid will provide a set of systemwide services for storage, retrieval, and management of data and metadata from experiments and simulations. The services, built on NMI protocols and extensions to NMI protocols, will interface to central data storage resources as well as data resources located elsewhere on NEESgrid, such as at sites. The data services and their interfaces to central data storage constitute NEESgrid's "central data repository." In addition, external data resources will be made interoperable with NEESgrid repositories through proxies that implement NEESgrid data protocols on the NEESgrid side and legacy protocols on the external side.

Interoperability of heterogeneous data and metadata will be achieved by community convergence on a set of recommended data and metadata models and formats and also by the use of mediation code developed by the NEESgrid SI and sites to translate between nonstandard data and metadata formats and NEESgrid-supported recommended standards. Data services will also accept data and metadata in nonstandard formats but will not provide as sophisticated retrieval and access capabilities for these data types.

The SI will provide storage resources during the development phase sufficient to store and retrieve data from complete experiments from participating sites. The SI-provided storage resources will include high-performance, redundant file systems accessible through NMI and NEESgrid data protocols, as well as a high-performance DBMS system for metadata indexing. Video and audio data will be stored, indexed, and retrieved using these same mechanisms. The central data resources will be located at NCSA and accessible over gigabit Ethernet. Initial storage resources accessible through NMI services will go online during the Early Adopter Program, with additional storage resources going online during the development of repository services. In the long term, these storage resources and repository services will be transitioned to the consortium.

5.2.1 Data Repository

A critical NEESgrid component is the distributed data repository, used to contain both experiment data and metadata. Architecturally, this component consists of

- local storage resources at equipment sites,
- centrally managed storage and indexes managed by the SI and later by the consortium, and

- a set of protocols and services connecting the first two components with each other and with NEESgrid users.

Data and metadata generated at equipment sites will be gathered from local storage and transferred to the central repository using GridFTP [Allcock] ([1] and www.globus.org/data) and a NEESgrid Metadata Harvesting Protocol (NMHP) using a similar strategy as the Open Archives Initiative Protocol for Metadata Harvesting [9]. Mediation codes developed or adapted by the sites perform any conversions necessary to bring the variety of site-specific formats into compliance with NEES-wide data and metadata standards, although the repository accepts data in any format. NEESgrid storage resources which are stored in systems other than file systems, such as relational databases, will be made accessible to NEESgrid through proxies that adapt NEESgrid data protocols to their API's.

Users can retrieve data from the central repository using the NEESgrid Data Discovery Protocol (NDDP), which allows datasets to be located based on attributes of interest. The functionality of this protocol roughly mirrors that of Information Retrieval protocols such as ISO Z39.50 or the Simple Digital Library Interoperability Protocol and also contains additional provisions for Grid-based security and data access, based on GSI and GridFTP.

External data resources are made transparently available through a proxy architecture in which a mediation component translates NDDP and GridFTP requests into a legacy protocol, such as a CORBA, ODBC, or HTTP/CGI.

The use of Grid services such as Globus Replica Location Service (RLS) allows data to be stored at the optimal location for a particular application without requiring clients to update references to each data object. For instance, a site might frequently access its own data from the repository, so that data might be replicated locally; the RLS allows this replication to be performed transparently to the applications that need to access the data.

The aim of the architecture is to provide a set of services and protocols that are as consistent as possible across all the NEES sites and data resources.

5.2.2 Schema for Data and Experiment Metadata

In order for researchers to be able to make sense of the data and metadata stored in the repository, that data and metadata must be stored according to standard schema. The NEESgrid SI team is working with the NEES sites to produce recommended data and metadata standards to support the kinds of data being produced within the EE community. The process of developing these standards is being driven by scenarios of data production and use, which are currently being developed and gathered. Drafts of these standards are scheduled for completion in 2Q 2002 and will be distributed for comment. Because it is impossible to anticipate all possible use scenarios, the repository recommends, but does not require, that data be represented in a single or small set of standards. NEESgrid repositories will accept this standard format as input. To deal with heterogeneous data formats, repositories will employ components for translating data from one format to another. Data and metadata are represented in portable, extensible formats such as XML and RDF in addition to existing formats so that translation components can be rapidly developed and adapted using off-the-shelf tools.

One of the central NEESgrid elements is the distributed data repository, which will contain experiment data and metadata. This will consist of some number of storage servers, on which the actual data and metadata live, and catalogs to locate physical copies of files. We have chosen this design (rather than a centralized storage server) to provide enhanced performance (sites can choose to keep copies of some data local and may even choose to use the same machine as their

NEES-POP and a local data repository) and reliability (researchers can get at local copies of files even if they can't reach a remote server).

To move data to and from the data repository, we will use the GridFTP protocol; this consists of extensions to the standard FTP protocol to support GSI security and enhanced performance and reliability.

5.2.3 Data and Metadata APIs, Tools, and Procedures

The NEES SI will provide tools and API's for generating, managing, locating, and accessing data and metadata. These will consist of:

- *Data and metadata "publishing" APIs and procedures*. The NEESgrid SI will provide API's and procedures for transferring their data to the central repository. These procedures will include simple mechanisms such as placing data files in a local directory on the NEES-POP designated as an "outbox," as well as sophisticated mechanisms such as APIs for communicating with the repository using the NEESgrid data protocols.
- *Data and metadata mediation components and APIs*. To maximally interoperate with other NEESgrid data, site data will need to be encoded in NEESgrid-recommended data and metadata models and formats. In order to facilitate this, the NEESgrid SI will work with sites to develop and adapt mediation codes that can translate existing data formats into NEESgrid-recommended formats.
- *Data and metadata management tools*. Maintaining data collections such as the NEESgrid repository requires administrative tools for adding, deleting, moving, and linking data items. The NEESgrid SI will provide basic tools of this sort for the NEESgrid repository. These will enable NEESgrid users to manage data items as well as associating metadata with data items or other metadata items (annotation and linking). The tools will also allow users to edit authentication and authorization access control information associated with data items.
- *Data location tools*. Tools for locating data items based on metadata attributes will be provided by the NEESgrid SI, although the tools will have limited features. A full-featured user interface tool, specialized for the variety of data discovery tasks required by users, is beyond the scope of the SI's infrastructure-development effort. However, the SI will provide an example application exercising the functionality of the NDDP; this application's code can be reused in building full-featured user tools.

5.2.4 Central vs. Local Repositories

NEESgrid repository services will be based on NEESgrid data protocols, which can be interfaced to a variety of storage resources, enabling various levels of repository functionality for those storage resources. Thus, sites could use NEESgrid-provided repository software to maintain local repositories based on local data resources such as file systems and databases. The functionality of these local resources would be limited by the functionality of the underlying data resources; for example, if the local data resources were not redundant and not backed up, the local repository would not be as reliable as the NEESgrid central repository. But sites could also choose to maintain a local repository in order to provide capabilities in addition to what the NEESgrid central repository provides, such as low latency or specialized indexing and location capabilities.

5.3 Information Services

We will provide the capability to discover the existence and properties of resources, such as compute servers (e.g., for simulation purposes) via the Globus Meta Directory Service (MDS); we will provide new MDS back-end processes to enable discovery of additional resources such as shake tables, instruments, and simulation code repositories. See Figure 10.

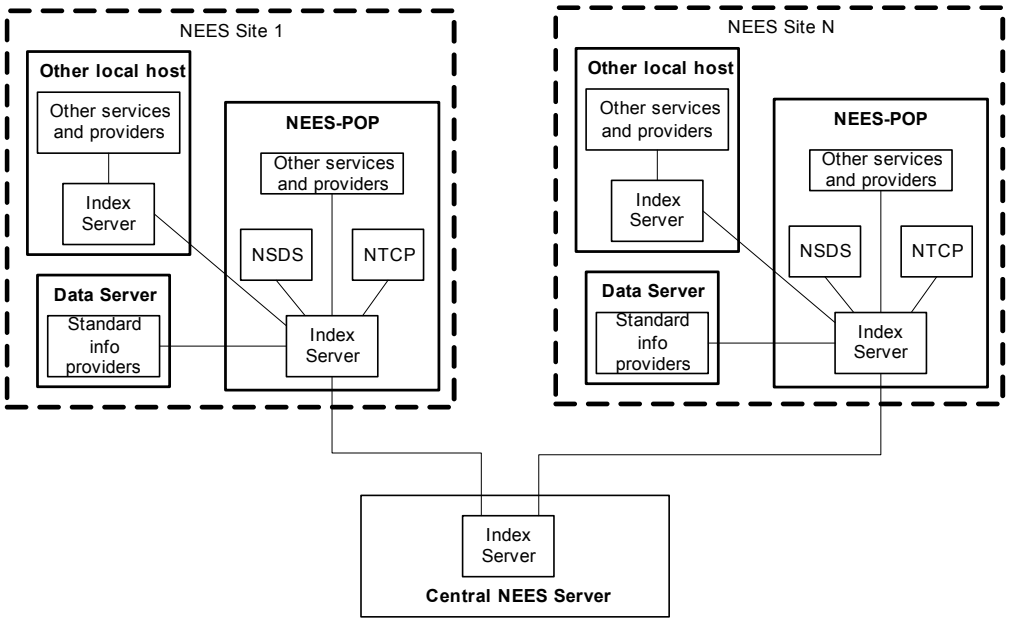

Figure 10: NEESgrid resource discovery architecture

Information services will play an important part in the NEESgrid. While a general definition of information services might include many aspects of data management such as data discovery, data retrieval, and data storage, this section limits the definition of information services to the capability to discover the existence and properties of NEESgrid resources such as the NEES-POP hosts, site instruments, and other site computational or storage resources. The information services component of the NEESgrid will be implemented using the Globus Toolkit Meta Directory Service (MDS). Other aspects of data management are described in other sections of this paper.

5.3.1 Components

NEES POPs run standard NEESgrid "middleware" services that link site resources into the national-scale NEESgrid collaboratory. The Information services middleware component provides the capability that allow remote users to determine the characteristics and status of site resources, including availability, software versions, and tool location, with authentication and access control if desired. These services can also provide access to performance information on networks and local resources, obtained via passive performance sensors and/or active network performance tests.

The Information services model for the NEESgrid will be implemented using the virtual organization concept as supported by the MDS. This concept allows the formation of relationships between resource and index servers at one or more sites. In the NEESgrid model, resource servers on each host will report to an index server on that host, which in turn will report to an index server running at the site's NEES-POP. The index servers at each site's NEES-POP

will report to the top level NEESgrid index server. This approach addresses scalability and administrative issues by providing a straightforward mechanism to integrate new sites' information services into the NEESgrid without any impact on existing sites and without any administrative overhead for the new site. That is, when a new site joins the NEESgrid, the Information services software deployed on the site's NEES POP will be configured in such a way as to automatically register itself with the NEESgrid Information services top-level index server.

The function of the index server on each host is to collect information from the various services and other information providers on that host (for example, a user may query a host's index server to find out whether a streaming data server is running on that host, and if so, how to contact that server; the user would then most likely send additional queries directly to that streaming data server). The function of the index server on a site's NEES POP is to aggregate the information collected by all information providers that are affiliated with that site (for example, a user might query that index server to find information about all file servers at the site available for use by NEESgrid users). The NEESgrid top-level index server then collects information about all NEESgrid site (for example, a user might query the top-level index server to find the NEES-POP associated with a particular institution, and then query that NEES-POP's index server for attributes of a particular instrument located at that site).

5.3.2 Data Objects

A schema consisting of object classes and their corresponding attributes describes the data represented in the MDS. A number of commonly used object classes representing computing resources have already been defined and will be available via queries to the NEES POP's index server. Examples of the data these objects represent include hardware characteristics, software versions, and system load. Information represented by these types of object classes is dynamic. The information is collected by a set of information provider programs or scripts that are invoked periodically to provide up to date information. Other types of object classes that represent static data can also serve a useful purpose. An example of these types of object classes is information about the simple existence of an instrument. The attributes of such a static object class could then provide specific attributes of the instrument. Since information for these types of object classes is not collected dynamically; an initial object class must be defined, and its information must be made available to the particular server that will be responsible for servicing queries about the object.

5.3.3 Security

The MDS supports both authentication and authorization capabilities. The authentication mechanism, Grid Security Infrastructure (GSI), is the same authentication mechanism used by the other middleware services. Access control for service data is provided by the individual services providing that data.

5.3.4 Data Acquisition

While the basic configuration of the index servers is performed at the time of software deployment, any additional configuration changes (such as the addition of new information providers) must be made by hand. As part of the middleware services, however, a set of Unix command line client tools have been developed that facilitate obtaining information from index servers (and service data from resource servers). These tools allow the user to make queries and specify common query characteristics such as filters. In addition to these tools, a set of JAVA tools has been developed and used in Web portals, to provide users with a Web-based interface to the MDS.

5.4 Telepresence Services

NEESgrid provides a set of standard interfaces for configuration, observation, and control of experimental facilities. To facilitate the creation of general mechanisms that can be used to observe and control a wide range of experimental facilities, we define a generic high-level model that can be used to represent a wide range of different experiments. In this model, an experiment is characterized by

- a set of *control points* that provide network interfaces for observation of experimental data and actuation of experimental apparatus,
- an arbitrary number of descriptive elements that describe the experiment being performed, and
- a sequence of one or more *trials* that result in data being made available on a specified data repository.

We describe each of these elements in more detail below.

Each experimental configuration defines a set of control points, which are software abstractions that provide network interfaces to data acquisition and actuator configuration channels. An experimental control point that is used only to acquire data is called a *sensor*. Each experiment will have a number of physical measurement devices (e.g., strain gauges, accelerometers, displacement gauges) which produce a stream of data that is captured by the data acquisition system. We can associate a sensor with this data stream. A remote client can then *subscribe* to that sensor, and the client will receive experimental data is it is generated in near real time subject to network and resource constraints. Streaming sensor data is distinguished from experiment result data in that a subscription to a data stream provides only future data values (not record past data values) and that access to sensor data may be limited for resource management reasons (for example, network bandwidth or latency between the requestor and the sensor). Of course the ability to subscribe to a sensor is subject to access control, community (i.e., experimental) and local policy (see Section 5.1). Thus, not all sensors associated with an experiment are necessarily available to all potential clients, or even all participants in an experiment.

Control points that are not sensors support the setting of parameter or command values. A control point represents an abstraction of a teleoperation interface, and not necessarily a direct connection to a physical device. Thus, sending a network command to an actuator may result in a specific activity in a control system (for example, “set displacement-in-x-direction of control point A to 1cm” may result in an interaction with a physical actuator), or it might pop up a message on the operators console requesting that an experiment be manually initiated (much like the tape mounting command on older mainframe computers).

One need not have a sensor for every data-channel associated with an experiment, nor does one need a control point for every actuator. As part of the design of the experiment, one may decide to enable only a subset of the data acquired to be available “immediately” via the streaming data interface, or one may decide that there is no need to allow remote users to control a particular actuator. Conversely, there is no requirement that a control point must correspond to a physical device: a control point may correspond to a set of several actuators (for example, “rotate control point A 10 degrees on the X axis” may result in two actuators being activated), or, for high data-rate data channels, it might be desirable to have a sensor interface provide data streams that contain only a subset of the measured data. Sensors that combine physical data channels may also be desirable. In fact, there is no requirement that *any* physical actuators or sensors be involved at all – the same protocols and services used for physical control can be used to control

computational simulations (both to facilitate hybrid experiments, and to enable sites to "sanity-check" an experiment using a computational experiment before committing the resources to performing a physical experiment). Because the network interface to an experiment is not defined by its physical configuration or local software interfaces, the control point abstraction provides NEESgrid teleoperations services with a high degree of flexibility.

A range of information may be required to interpret experimental data. This can include references to CAD drawings, sensor inventories, and calibration factors.. We refer to this experimental metadata as *descriptive elements.* (We avoid using the term descriptive elements to avoid confusion with metadata associated with data sets. We note, however, that descriptive elements may become part of the metadata associated with the experimental data.) The descriptive elements will vary from experiment to experiment and from facility to facility. Descriptive elements may be made available via the index service that is resident on the NEES POP and will be stored as part of the metadata associated with experimental trials.

An experiment consists of a sequence of one or more *trials.* As a result of performing a trial, all collected data, descriptive elements, and associated trial metadata must be stored on at least one NEESgrid-compliant repository where they can be accessed via the protocols, tools, and interfaces defined in Section 5.2. The decomposition of an experiment into trials will depend on the nature of the experiment. On short-duration experiments such as on a shake table, each input signal may result in a separate trial. On longer-running experiments, a new trial may be initiated every time a control point is provided with a new value, or trials may be defined on specific time boundaries. What is important is that access to the complete data set is guaranteed only at the conclusion of a trial.

The NEESgrid experimental model has been designed to be general enough so that it places few restrictions on how a specific experiment is mapped to it. Even so, the model provides enough structure so that essential elements of all experiments can be supported as a fundamental part of the NEESgrid system architecture.

5.4.1 Data Acquisition Procedures

The telepresence interface provides near-real-time monitoring of a possible limited subset of the actual or virtual data channels that are represented via sensors in a particular experimental configuration. Access to actual experimental data is provided via NEESgrid data repository interfaces, however, and only at the termination of a trial. Thus, the telepresence interface can be used to configure an experiment, to determine what repository the data will land on, to initiate the experiment, and to provide the ability to observe specific data channels. The implementation of the telepresence system must arrange for the transfer of the resulting experimental data from the local data acquisition system to the specified repository. For reasons of performance and robustness, in most instances the target repository will be the local repository associated with the experimental site.

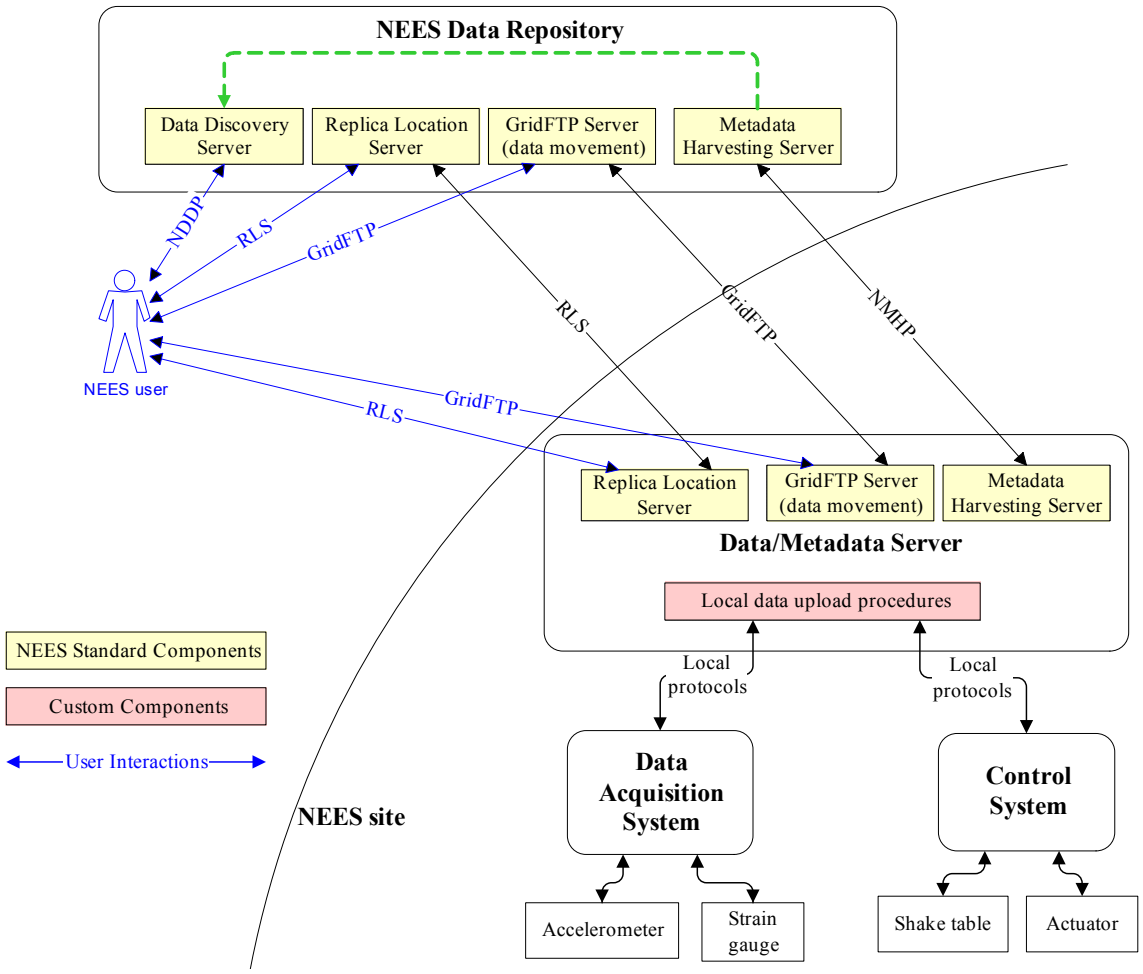

Figure 11: Data acquisition and conversion in NEESgrid

The advantage of this approach is that the details of underlying data acquisition system can be hidden from the end user with little or no loss of functionality. We also have access to the rich range of functions supported by the NEES data repository including community based access control, attribute based discovery and high-performance, reliable transport of data from the experimental site to other NEES data repositories, including the central data repository.

An overview of the data flow in the NEES data acquisition cycle is shown in Figure 11. Users access experimental data via the NEESgrid data management protocols (GridFTP, RLS, NMHP). Data is moved from the data acquisition system and staged onto a storage repository by components of the NEESgrid telepresence system that execute on the NEES-POP. The method used to stage experimental data from the data acquisition hardware to the repository depends on the characteristics of the data acquisition system and will vary from site to site. For data acquisition systems that leave data on a local, shared file system, configurable, generic transfer code can be used to move the data to the local repository. Interfacing more exotic data acquisition systems may require that the site write specialized code to perform this transfer under direction of the telepresence services running n the NEES POP.

An important element of a NEES data repository is the association of metadata with raw data files. In the case of data acquisition, the experimental parameters are an important element of this metadata. Thus a means is needed for extracting configuration information from the control

system and associating it as metadata of the experimental data. As with data staging, metadata extraction and publication are performed by elements of the telepresence service running on the NEES POP. Again, this may require development of site-specific code that knows how to interface with the particular control system being used for the experiment.

To simplify the development of site-specific data acquisition and control system interfaces, the telepresence service will have well-defined interfaces into which local access functions can be "plugged.". In addition, the telepresence library will provide utility functions to aid in operations such as data formatting, transfer of data to a data repository, and publication of metadata. Thus, while site-specific code is not part of the NEESgrid system architecture proper and will be the responsibility of equipment site, we estimate the complexity of this site-specific code to be low, 0n the order of a thousand lines of code.

5.4.2 Control Protocols and APIs

A more detailed view of the telepresence service is shown in Figure 12, which illustrates the interfaces and software that would be used to implement a pseudo-dynamic test in which an application controls components in a physical experiment.

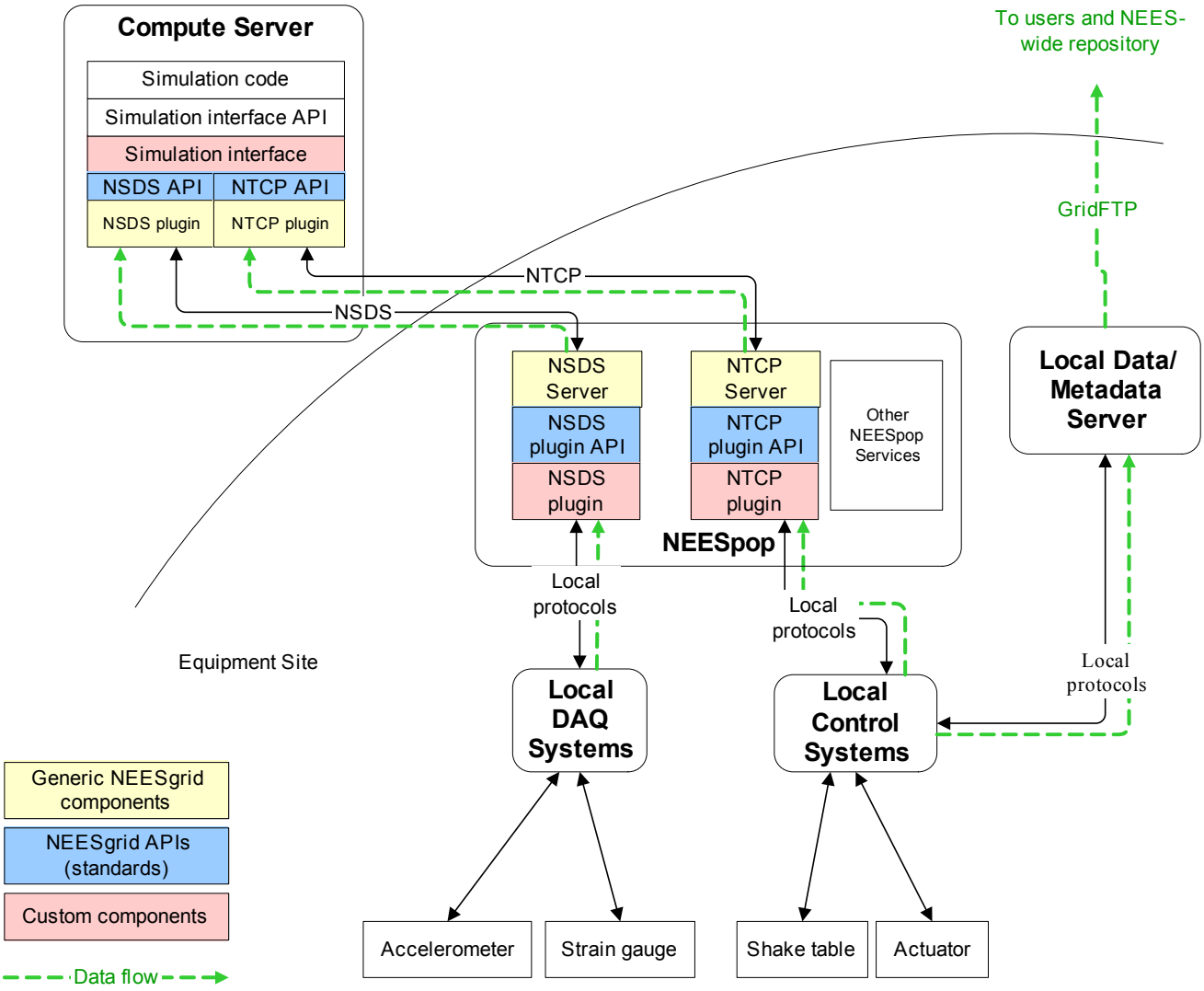

Figure 12: Structure of NEESgrid telepresence services.

As illustrated in Figure 12, the NEESgrid architecture will include APIs and protocols for telecontrol operations, including a control API and protocol that will define calls to act on control points (read data from an instrument, control an actuator, etc.).

Part of the software installed on the NEES-POP implements the telepresence service. Operations supported by this service will include subscribing to a data steam from a sensor, sending control operations to a control point, and various management functions such as specifying where experimental data should be placed on the data repository. As illustrated in the figure, the streaming data service (NSDS) has three components:

- A standard streaming data server that implements network protocols, parses remote commands, enforces access control requirements and parses remote commands and dispatches the to local data acquisition operations via the client API. Initial versions of the server will provide straight data pass-through with simple access control. The

functionality of the service will be expanded during the development cycles to support server side data reductions and more sophisticated access policies.

- A streaming data service driver API that defines the interface between the generic NEESgrid streaming data services and site specific streaming data drivers.
- Equipment-specific drivers that implement the telepresence service API defined above and provide the interface between local data acquisition hardware.

Similarly, the telecontrol service (NTCP) has these components:

- A standard telecontrol server that implements network protocols, parses remote commands, enforces access control requirements and parses remote commands and dispatches the to local data acquisition operations via the client API. Initial versions of the server will provide transaction-based control operations and simple access control.
- A driver API that defines the interface between the generic NEESgrid telecontrol services and site-specific telecontrol drivers.
- Equipment-specific drivers that implement the driver API defined above and provide the interface between the telecontrol server and local control systems. Depending on a site's configuration and safety requirements, the site-specific control commands may be something as simple as sending a message to a (human) equipment operator.

In addition to control and observation functions, an equipment-specific driver stages data onto a data repository and extracts experiment parameters and publishes them as metadata.

Basic interactions with the telepresence are performed via the NEES Streaming Data Service protocol (NSDS) and the NEES Teleoperations Control Protocol (NTCP). These protocol will be based on SOAP over HTTP, making it compatible with emerging Web services. Moreover, some sensors may use additional protocols optimized for specific data types, such as streaming audio and video.

The NEESgrid software release will include a client library to facilitate integration of telepresence functions into NEES applications. Figure 12 shows how the telepresence client library could be integrated into a simulation code to construct a pseudo-dynamic test. In addition to the programmatic API, the NEESgrid software release will include command line tools and browser-based interfaces for constructing Web-based telepresence interfaces.

5.5 Simulation Support

The simulation component of NEESgrid is based on one ambitious goal: the demonstration of the utility of computational simulation in earthquake engineering, including using simulation to augment individual physical experiments, to coordinate distinct physical experiments, to optimize the value of special aspects of physical experiments, and to replace physical experiments altogether in some cases. The underlying computational subsystems of NEESgrid are designed to facilitate achieving this ambitious goal over the lifespan of the NEES consortium.

There are three related strategies in NEESgrid that support this simulation goal:

1) development of an earthquake engineering simulation portal that will permit the community of researchers and practitioners to find and utilize computational simulation tools, content obtained from such tools, and documentation to enable the productive use of such tools or content,

2) collection and dissemination of software and results required to populate the simulation portal, from both community-oriented (i.e., open-source) and from proprietary venues, and

3) coordination efforts between NEESgrid and related computational simulation efforts already in place within the earthquake engineering community, so that the advantages of the NEESgrid system can be realized by these ongoing community efforts, while these simulation projects can be used to demonstrate the utility of computational solutions to earthquake engineering problems.

These strategies are used to derive the underlying simulation subsystem for NEESgrid, as shown in

Figure 13 below.

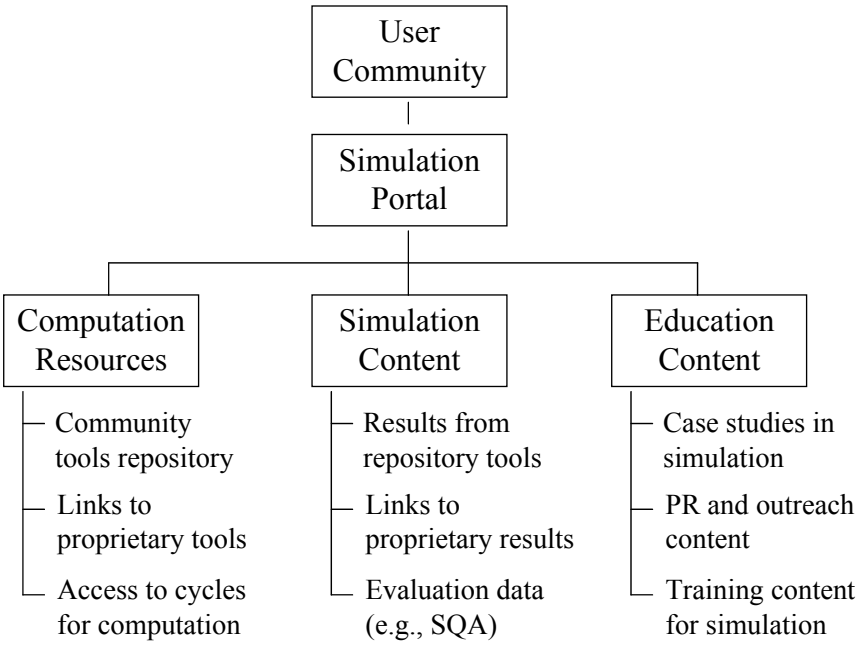

Figure 13: Structure of NEESgrid simulation portal and subsystem.

The portal provides for uniform access to simulation tools, simulation results, and education and public-outreach content. This access is supported by various capabilities that augment the value of the community portal, including searches based on data and metadata information, and computational resources where community software tools can be located, obtained, and effectively utilized.

In summary, the computational subsystem architecture of NEESgrid will superficially resemble the overall architecture of the shared-use equipment site subsystem that forms the nucleus of NEES, but with computational capabilities augmenting and/or replacing the diverse experimental capabilities of the various equipment sites.

5.5.1 Computational Resources

Resources that support computational simulation under NEESgrid include:

- A code repository of open-source or other redistributable tools that facilitate computational simulation in earthquake engineering, e.g., computational frameworks such as components for PEER's OpenSees application environment, community-based computational modules that enhance commercial programs (e.g., UMAT routines for utilizing ABAQUS in earthquake engineering practice), systems identification toolkits based on MATLAB, etc.
- A library of references (e.g., links, sample tools, etc.) that permit access to other software tools that cannot readily be distributed via the NEESgrid computational simulation repository, e.g., commercial tools, open-source tools that already possess working code repositories, and
- Access to computational simulation sites that will function in the same manner as NEES equipment sites (i.e., as shared-use facilities supporting the work of the NEES consortium), but whose function is virtual, e.g., providing cycles for use with community tools such as OpenSees, providing archival capabilities for storage of larger simulation datasets that may not be desirable within the NEESgrid centralized data repository, etc.

Appropriate grid protocols will be utilized to support these computational resources, e.g., GridCVS for remote authenticated access to the community tool repository. The computational resources will be subject to the same flavors of discovery mechanisms used for the experimental sites, e.g., metadata searches appropriate for capabilities. In addition, browsing and searching the computational resources will include computation-specific capabilities relevant to the practice of civil engineering, e.g., classification of resources via adherence to software quality criteria, support for codes of practice, and other earthquake-engineering-specific requirements appropriate for software use.

Computational capabilities will be augmented using grid services that will permit remote job entry and monitoring of computational tasks on various compute servers used with NEESgrid, or grid services to facilitate computational workflows, e.g., coordinating a computational simulation process on a remote cluster with a rendering process on another cluster in support of visualizations displayed that can be used to monitor progress of the original computational simulation.

5.5.2 Content Resources

The content provided within the computational subsystem of NEESgrid will parallel the architecture provided for the computational resources, but with simulation results replacing software tools. Because of the complexity of the physics associated with many branches of earthquake engineering (e.g., soil liquefaction, soil-structure interaction problems, etc.) and the great uncertainties present in characterizing the data required for computational simulations in earthquake engineering (e.g., material information, earthquake characterization, etc.), the field of earthquake engineering has not been able to utilize computational simulation techniques as readily as related engineering fields. In order to facilitate the use of computational simulation within earthquake engineering, it is necessary to provide content that demonstrates the value of computational techniques, and to provide educational content that aids researchers and practicing engineers in developing and deploying computational solutions to earthquake engineering problems.

The content resources of NEESgrid's computational subsystem will permit finding and using results from a variety of community and commercial software tools, so that practitioners, researchers, and education/outreach personnel in earthquake engineering can insure that they fully

understand the computational capabilities of NEESgrid. As in the rest of the NEESgrid computational subsystem, the architecture that supports utilizing the content resources will closely resemble that required to utilize experimental data.

5.6 Administrative Grid Services

A number of administrative services will be required to turn NEESgrid into a production facility. Management of user accounts, Grid operations, software version control, and the support desk are the essential administrative services. In this section we discuss these administrative services in detail and describe how they will be managed.

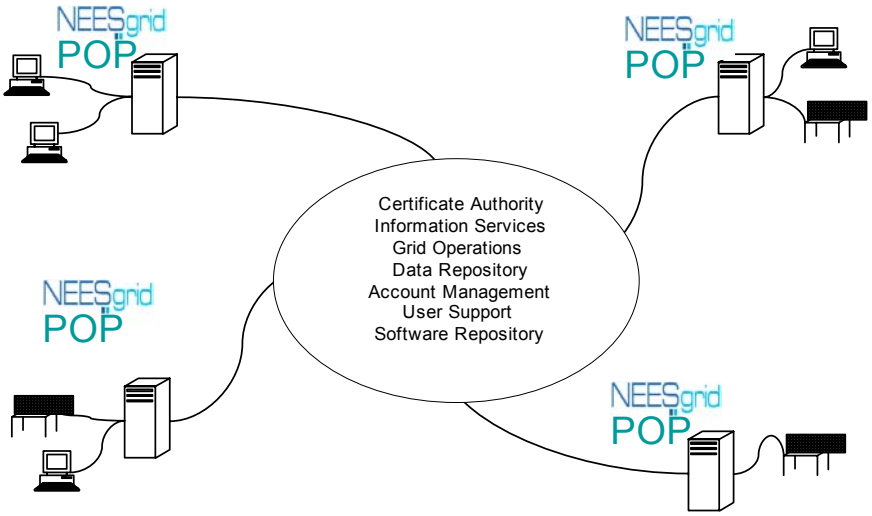

Figure 14: NEESgrid administrative functions are structured around robust, centrally operated administrative functions.

As shown in Figure 14, administrative Grid services run behind the scenes to deliver a coordinated, reliable, standardized environment for both the Grid resource owners and Grid users. NCSA and the Alliance already have a well-established set of administrative services in production, which we plan to extend to include the NEESgrid project. NEESgrid will appear as an Alliance project, giving NEESgrid a 24/7 operations center for monitoring and problem reporting, trouble ticket system for problem tracking, user consulting for front-line support, Certificate Authority, and a complete user account management.

5.6.1 NEESgrid Account Management

The account management service coordinates the management of user accounts throughout the NEESgrid. Specifically it creates, modifies, and removes user accounts across the NEESgrid resources in a uniform and coordinated manner. This is an onerous task that needs to be done in order to create a consistent user-friendly environment. It was overlooked in the early planning phases, however, and hence the service is unfunded. We proposed to leverage the centralized NCSA Alliance account management system by extending it to the NEESgrid project and push account management transaction to NEES POPs. This is still an open issue, however.

This service must include traditional systems account management plus Grid certificate generation, briefly mentioned above in Section 5.1.2,, management of Grid certificate mapfiles, and certificate revocation lists. Grid certificates as described above are used to authenticate users to NEESgrid resources. The mapfile maps Grid certificate identities to local user accounts. With

this method we are able to provide NEESgrid users with single sign-on to NEESgrid resources in a secure and scalable way. Certificate Revocation Lists (CRLs) are used to revoke authentication certificates that are no longer valid. Certificates are revoked typically because a user has either lost the key (think password) or because one suspects that an account has been compromised. Through the use of a coordinated management of NEESgrid accounts, we will be able to create, modify, or remove user accounts across NEESgrid in an automated way. Project PIs will be able to request user accounts to a central service, and accounts will be pushed out to the requested NEES POPs, saving the PI from making multiple account requests for each NEES POP.

Major NEESgrid account management functions include the following:

- Coordinated account generation across NEES POPs and central services.
- NEESgrid authentication certificate generation
- Coordinated certificate revocation service
- NEESgrid user registration
- Grid map file management

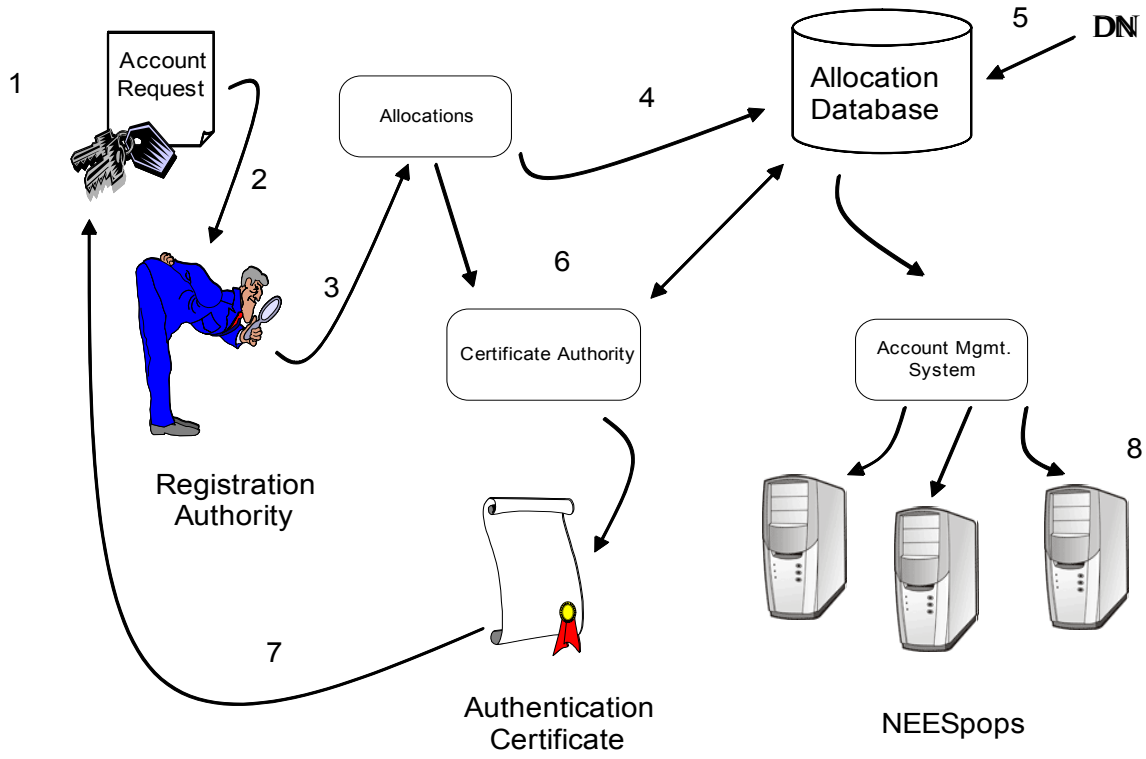

Figure 15: Overview of NEESgrid account creation procedure

The sequence from account request to account creation is illustrated in Figure 15 and described below:

1. Principal Investigator (PI) makes project allocation request.
2. Once the project has been approved, the PI account is created and project collaborators can request accounts for the project (must be approved by PI).

3. Collaborator accounts are created and the requestors are notified.
4. Users request X.509 certificates from the NCSA certificate authority (CA) via certificate request tools from their site neespop system.
5. CA validates requests, generates X.509 certificate, and notifies requestor.
6. User retrieves X.509 certificate for immediate use.

Modification and deletion of an account will be triggered via PI or user requests to the NCSA Alliance allocation group, which will trigger an automatic update across the NEES-POPs.

5.6.2 Grid Operations Center

Instrumentation and monitoring are important to NEESgrid operations as a means of identifying network failures and ensuring smooth operation of NEESgrid as a whole. Hence, our NEESgrid architecture places a high priority on monitoring NEES-POPs and collecting network performance measurements.

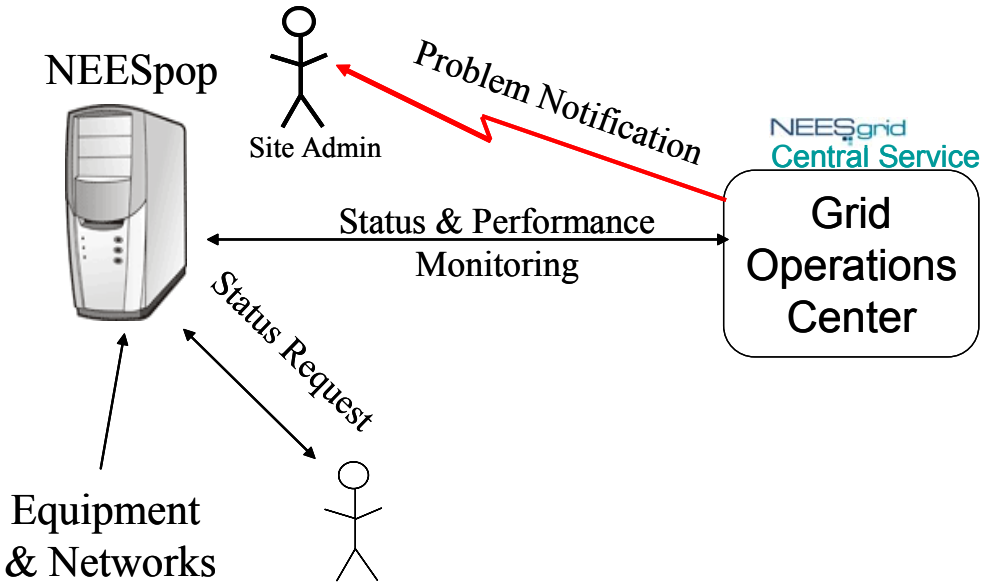

Figure 16: NEESgrid administrative structure for monitoring NEES sites

As illustrated in Figure 16, equipment sites will be monitored through software resident on the NEES POP. The Network Weather Service software, Globus resource information collection routines, Big Brother, and Iperf will be the primary monitoring tools deployed to the NEES POPs and monitored remotely by NCSA Operations. Resource up/down status will be audited through the Globus Meta Directory Service and/or Big Brother. In addition, the Network Weather Service will track network performance and report to MDS. NCSA operations will monitor NEES-POPs and network performance 24/7, identify potential problems, and work with NEESgrid equipment site administrators to resolve problems.

NEESgrid resources status information will be collected into the information service on the local NEES-POP. Special software will need to be developed jointing by the SI team and the equipment site to collect this resource status information. These will be simple scripts that verify basic availability of the resource via the network.

The Network Weather Service (NWS) running on the NEES-POP will collect local and wide area network performance measurements and register this data in the local NEES-POP for retrieval by NCSA operations, NEESgrid Users via the NEESgrid Portal, or , specialized NEESgrid applications.

NCSA Operations will monitor the status of NEESgrid resources and network performance by recovering the data from each of the NEES-POP information services and displaying this on an operations monitor. This same information could be made available to NEESgrid users via the NEESgrid User Portal. The monitoring software will be rigged with various alarms that, when triggered, generate an entry into the NCSA trouble ticket system. NCSA will coordinate problem resolution by working with local NEESgrid equipment site administrators.

5.6.3 Software Version Control

Software version control will be important to ensure that we present a uniform set of capabilities and documentation to the NEESgrid users. Three main components need to be supported: NEES-POP operating system and Grid software; NEESgrid client software, which will reside on the user's personal system; and a NEESgrid software repository that will provide a place to discover NEESgrid-specific software and configurations.

The SI team and the equipment sites will administer NEES-POPs jointly. Operations and deployment of NEESgrid grid software, based on the NMI software, will be coordinated by the SI team. The SI team at NCSA will test all operating systems Grid software configurations in-house prior to making them available to the equipment sites for installation. Perhaps all of the Grid software installations may be handled automatically; however, we are still researching requirements and capabilities of such a system. Since we will be using the NMI middle release as our Grid software base, we can concentrate on NEES-specific configuration issues, thereby l ensuring a uniform configuration across all NEES-POPs. Updates to the operating system will include timely security patches and bug fixes. After the early adopter program we anticipate no more than one major NEES-POP update every six months. During the early adopter program, updates may come more frequently. Site-specific additions to the NEES-POPs will be the responsibility of the equipment site. Since local configuration changes to the NEES-POP could dramatically affect consistency, reliability, and thus supportability, we plan to set policy with the goal of maintaining a consistent NEES-POP configuration across all of NEESgrid.

Client software and documentation will be updated as needed and posted to the NEESgrid software repository for download by NEESgrid users. The software repository will be centrally run at NCSA. This will essentially be an index to useful NEESgrid application software. The service will identify to NEESgrid users the appropriate versions, configurations, and documentation of specific application-level software. It will be accessible through a Web interface and the NEESgrid Portal.

5.6.4 Support Desk Services

The NEESgrid Support Desk will leverage the NLANR applications front-line support. The Support Desk will coordinate second-tier problems with appropriate NEESgrid staff. Services will be available via telephone, email, Web entry pages, and collaboration software such as CHEF. Consulting support on specific technical issues (e.g., use of NCSA supercomputers and code parallelization and optimization for high-performance platforms) will be handled by NCSA consulting services. All help requests will be tracked via NCSA trouble ticket-tracking system.

NEESgrid mailing lists will be managed centrally by NCSA.

6 NEES Services Transition Strategy

The NEES SI team is both developing new services and extending existing services at NCSA to support NEESgrid. These services include an information index server, centralized data repository, account administration, certificate generation, trouble ticket system, help desk, operations & monitoring, software repository, software integration and packaging, and the community authorization service. Here we describe how each of these services might be transitioned to another party at the conclusion of the SI award in 2004.

6.1.1 Index Information Service

The NEES Grid Index Information Service runs on a commodity platform and uses commercial and open source software. This service contains no unique information but rather is an index to other information services found throughout NEESgrid. As such, transitioning this service to another party is straightforward. Specifically it requires establishing a similar platform and operating system configured with the same open source Grid software, populating the index with the same information, and identifying this new index to the existing information services throughout NEESgrid.

6.1.2 Centralized Data Repository

The data repository for NEES as described earlier in this document will have a distributed architecture. We envision that data will move to the appropriate resting place, and in many cases we expect that place to be within the centralized data repository at NCSA. Transitioning this repository to another party would not be trivial, but the distributed architecture we have proposed will make it fairly straightforward. There would be significant effort in deploying such a repository at another site, given that NCSA's repository is operated by professional staff that take pride in the reliability and performance of the system and maintain it twenty four hours a day, seven days a week. The new organization would have to establish a new service, hardware, software, procedures, and operations. Data transformation routines might also be required if a different underlying storage system is used. We note that these were essentially the steps taken at NCSA and SDSC when the data from the Pittsburgh Supercomputer Center and Cornell Theory Center was moved to our archives.

The metadata catalogs will also have a distributed architecture, and much like the information service, any centralized metadata index could be transitioned readily.

6.1.3 Account Administration

Organizations typically have customized account administration procedures and tools. While most of the tools NCSA will use to manage the NEES accounts could be made available, it is unlikely that any organization would want to swap their existing solutions for NCSA. In addition, any new organization would likely be better off building a new customized solution to solve their specific needs rather than accepting NCSA legacy system. Nevertheless, some of the lower-level tools for direct management of the accounts could be made available. Taking over this service would require a significant effort on the part of the party.

6.1.4 Certificate Authority

The establishment of a Certificate Authority involves a significant amount of work. Some of the work is related to hardware, software, and configuration; however, a significant amount of work is involved in defining a certificate policy, staffing the service, and following procedures. We expect that by 2004 a significant number of organizations and universities will have established

their own certificate authorities and that a bulk of this service will no longer be required. We do not, however, envision that all users will be covered; therefore, another strategy will have to be implemented, such as use of commercial solutions.

6.1.5 Trouble Ticket Tracking System

Trouble ticket Systems are much like the account management system described above in that each organization typically creates its own unique procedures and tools. A number of commercial toolkits are available from which a system could be built. We assume that an organization capable of taking on such a service would leverage an existing system or build a new one. Since the user interface to this is email or the Web, the transition will depend on the new party's implementation of the service and procedures.

6.1.6 Help Desk

The help desk is standard. Hence, it and would be straightforward to transition the help desk to another party. The bulk of the transition would go into the training of support staff that would take over responsibilities.

6.1.7 Operations and Monitoring

We plan to use simple monitoring tools and capabilities that leverage open source software. As such, this software could be either transitioned or built from scratch with a small amount of effort. Since the data itself is transitional no data transfer tasks are associated with this.

6.1.8 Software Integration and packaging

NEESgrid software consists of a combination of custom software developed by the SI team, open source 3rd party software, and grid infrastructure software available from multiple sources. The deployment of this software requires integrations and configuration of these components dependent on how the software is to be deployed (i.e. some services can be run on the same host but may be distributed over several dedicated hosts to increase performance). A new release of software requires integration, configuration, and packaging of all components in order to produce a distribution that is easy to deploy. The procedures used to accomplish this task will be documented so that the responsibilities can be assumed by someone else.

6.1.9 Software Repository

The software repository is much like the Information Index Service and Metadata Catalog in that it is essentially open source software running on commodity platform that is an index to where the real data, in this case software and documentation, exists.

6.1.10 Community Authorization Service

Moving responsibility for the management of this service to another organization will involve training of appropriate staff but is not otherwise difficult.

7 Usage Scenarios

In this section, we present several scenarios to illustrate how the various NEESgrid components might be used.

7.1 Saving, Manipulating, and Retrieving Experiment Data

One of the simplest and most common usage scenarios is that a group of researchers runs an experiment and save the resulting data in a repository, from which it can later be retrieved—by those researchers or by others—for use in simulations, analysis, visualization, or other purposes

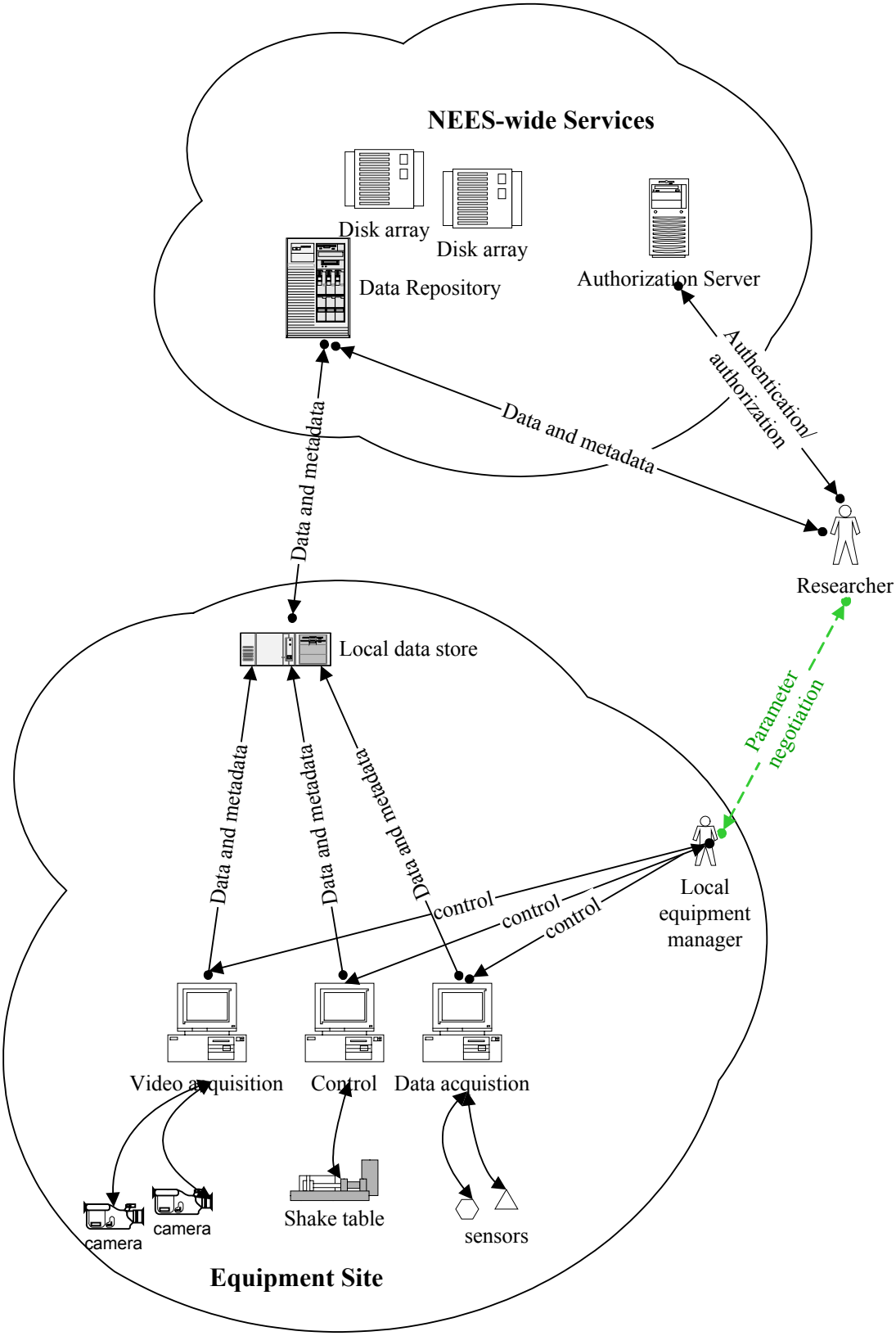

Figure 17: Saving, manipulating, and retrieving experiment data

.

In this scenario, a PI planning an experiment will use data and metadata "publishing" procedures described in Section 5.2.3 to add information (such as design drawings and descriptive text) about a planned experiment into the NEES data repository and will use the tools described in Section 5.1.3 to grant others (such as specific collaborators and graduate students) appropriate access to the data and metadata for that experiment. The researcher and other authorized users can then use the publishing and management tools to add annotations or additional data. At the same time, equipment managers will use those same tools to add calibration and other metadata regarding the actual sensors used.

Researchers and site equipment managers will then negotiate acceptable experiment parameters, and the equipment managers will run the first trial of the experiment. Raw data and metadata will be collected (data from instruments via the data acquisition system, calibration data from the control system, and video data from the video acquisition system) onto the local data repository, using the data acquisition procedures described in Section 5.4.1, and then published in the NEES distributed data repository using the data and metadata publishing procedures mentioned above. At this point, authorized researchers can retrieve the data and metadata from that first trial, perform some quick analysis and visualization, and confer with each other and the site equipment managers to determine the parameters for the second trial. This process will repeat until the entire experiment has been completed.

After an experiment (or a trial within an experiment) has been completed, a researcher may wish to do some data manipulation to filter out or correct "bad" data, for example, to invalidate data collected by a faulty instrument, or to specify a noise-reduction algorithm to be run on a particular analog data channel. Authorized individuals (that is, people to whom the person initiating the experiment has granted this access, as described above) will use the data management tools described in Section 5.2.3 to make these annotations and corrections.

Researchers will, of course, want to retrieve the data that has been saved in the repository. In many cases, some processing will need to be done to saved data to make it useful for the researcher; some examples of this are as follows:

- *Name conversion*: a researcher may, for example, be interested in values for an instrument with a particular symbolic name, but the repository may contain values for each data channel, with metadata mapping the symbolic names into channel numbers.
- *Data reduction*: a researcher may be interested in a time-series of acceleration values, but the data may be stored as time-series of voltages, with metadata specifying how the conversion can be done.
- *Synchronization*: a researcher may need to be able to correlate time values for instrument readings and video images.

The schema described in Section 5.2.3 will facilitate the development of code to perform these operations.

7.2 Monitoring an Experiment Trial in Progress

In the preceding scenario, researchers gained access to data and metadata from an experiment trial only after the trial was run (for example, a researcher might retrieve the results of a shake table run a minute after that run is completed). For longer-running trials, such as static or pseudo-dynamic experiments, researchers may wish to retrieve some of the data from that trial while it is in progress and view that data using a visualization tool or feed the data into a numeric

simulation, and then monitor that simulation's progress. For this scenario, the visualization or simulation software will be referred to as *client code*.

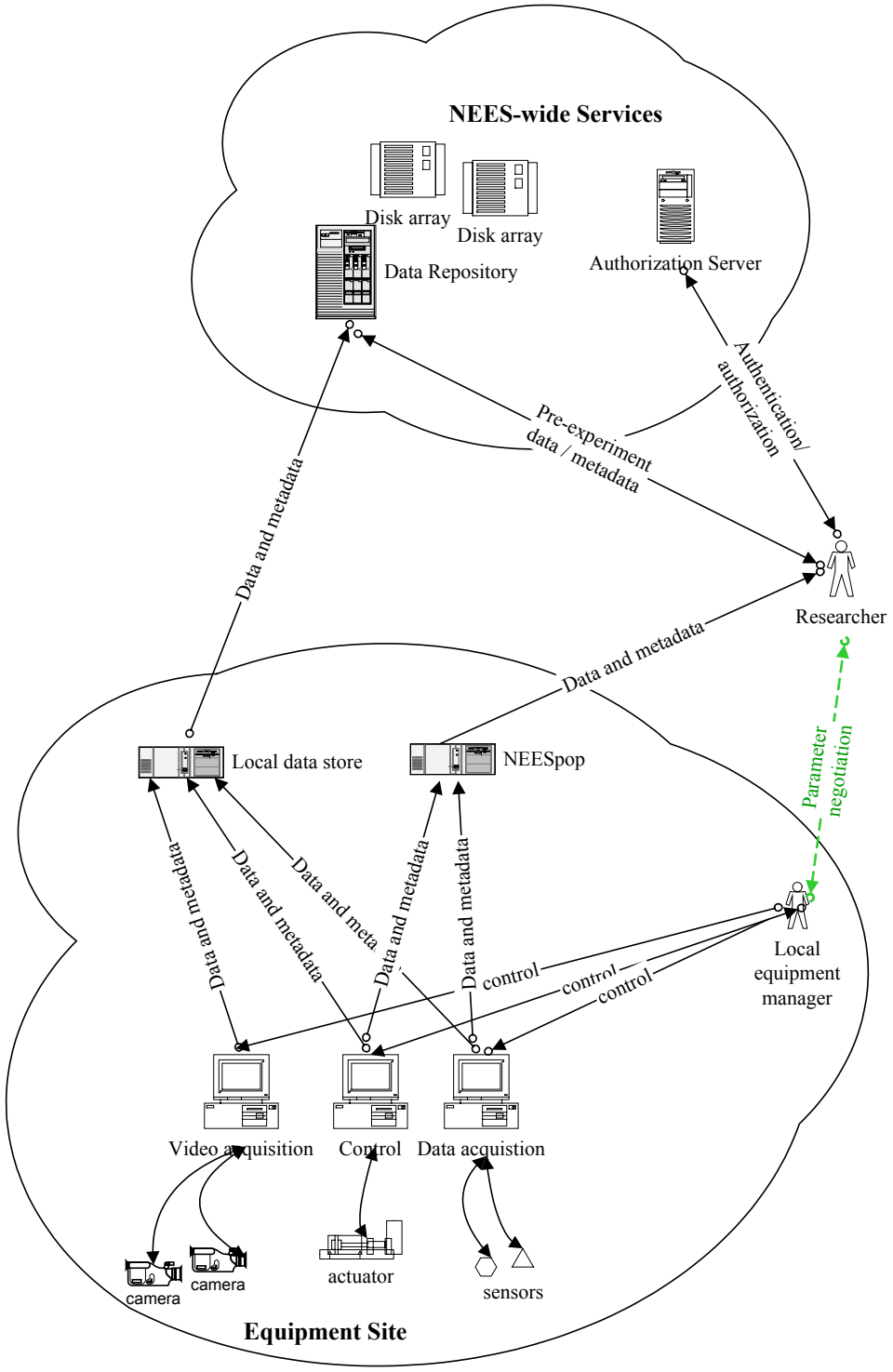

Figure 18: Monitoring an experiment trial in progress

In this case, the researcher will first make sure that the client code can accept input using the NTCP protocol described in Section 5.4.2. Depending on how the client code is implemented, the researcher may need to modify the code (or the underlying framework on which the code runs) to use the client API included in the NEESgrid software release.

The researcher will use the same processes described in Section 7.1 to enter descriptive data and metadata into the repository and will either start the client code on a local system (as shown in Figure 18) or use the standard Grid components described in Sections 5.3 and 4.4 to locate a suitable compute server and run the client code on that server. After this preparation is complete, the researcher will contact the equipment site to initiate the physical experiment trial. As the trial runs, the client code (or its underlying framework) will read streaming data from the NSDS server on the equipment site's NEES-POP. When the trial is finished, the researcher can enter the data from the physical trial into the repository using the methods described in Section 7; if appropriate, results from a numeric simulation may be entered into the repository in the same manner (using the data publication tools described in Section 5.2.3).

Security throughout the experiment is handled via the mechanisms described in Section 5.1. The researcher will use GSI to authenticate to the GRAM server when initiating the numeric simulation and delegate a credential, which the numeric simulation process will use when connecting to the site's NSDS server; standard GSI components will also verify the identities of the GRAM server and NSDS server. The site equipment manager (or another designated individual at the equipment site) will use the authorization tools to configure the site's NSDS server to accept "read" requests from that researcher for the duration of that experiment.

7.3 Evaluating the Feasibility of a Physical Experiment

An experimenter may wish to run a computational simulation as part of the process of evaluating a proposed physical experiment. The NTCP service (and appropriate drivers) can facilitate this by allowing researchers to "try out" an experiment by sending the same requests to a computational simulation that they plan to send to a physical experiment, as shown in Figure 19.

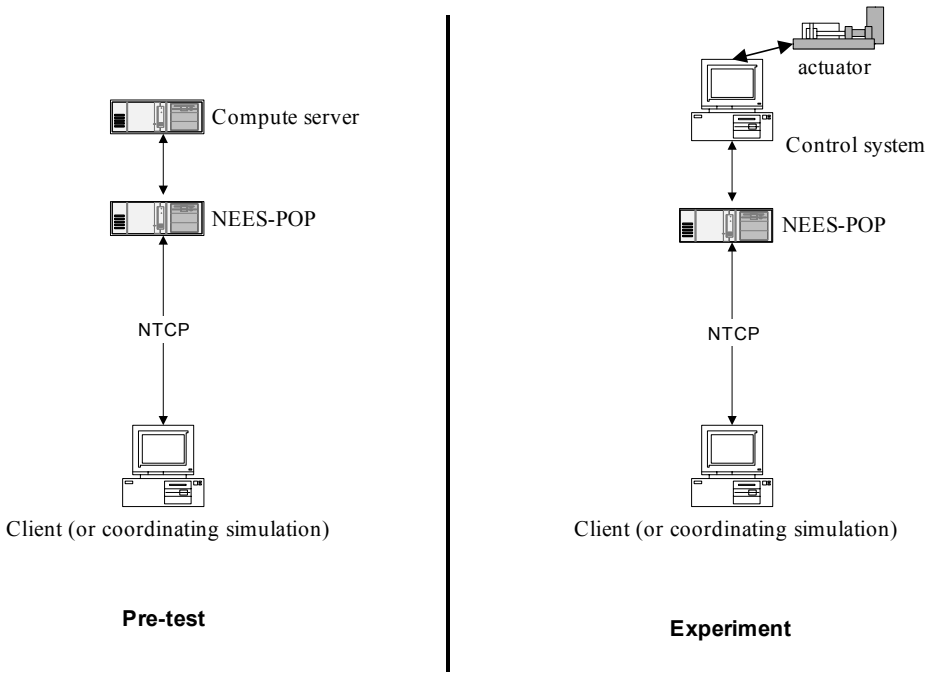

Figure 19: Pre-testing prior to a physical experiment

In this case, the researcher runs a pre-test by having a client application (or simulation) to send requests, via an NTCP server, to a computational simulation. The researcher can later use the same, unmodified client application (or simulation) to send the same requests, via an NTCP server, to a physical simulation.

7.4 Pseudo-Dynamic Experiment

Pseudo-dynamic experiments combine physical and numeric simulations; the results of the physical and numeric simulations at each time-step are used to determine the input (loading factor for the physical simulation, numeric input for the numeric simulation) for the next time-step. Some pseudo-dynamic tests are run slowly and do not have the stringent latency requirements that real dynamic tests have.

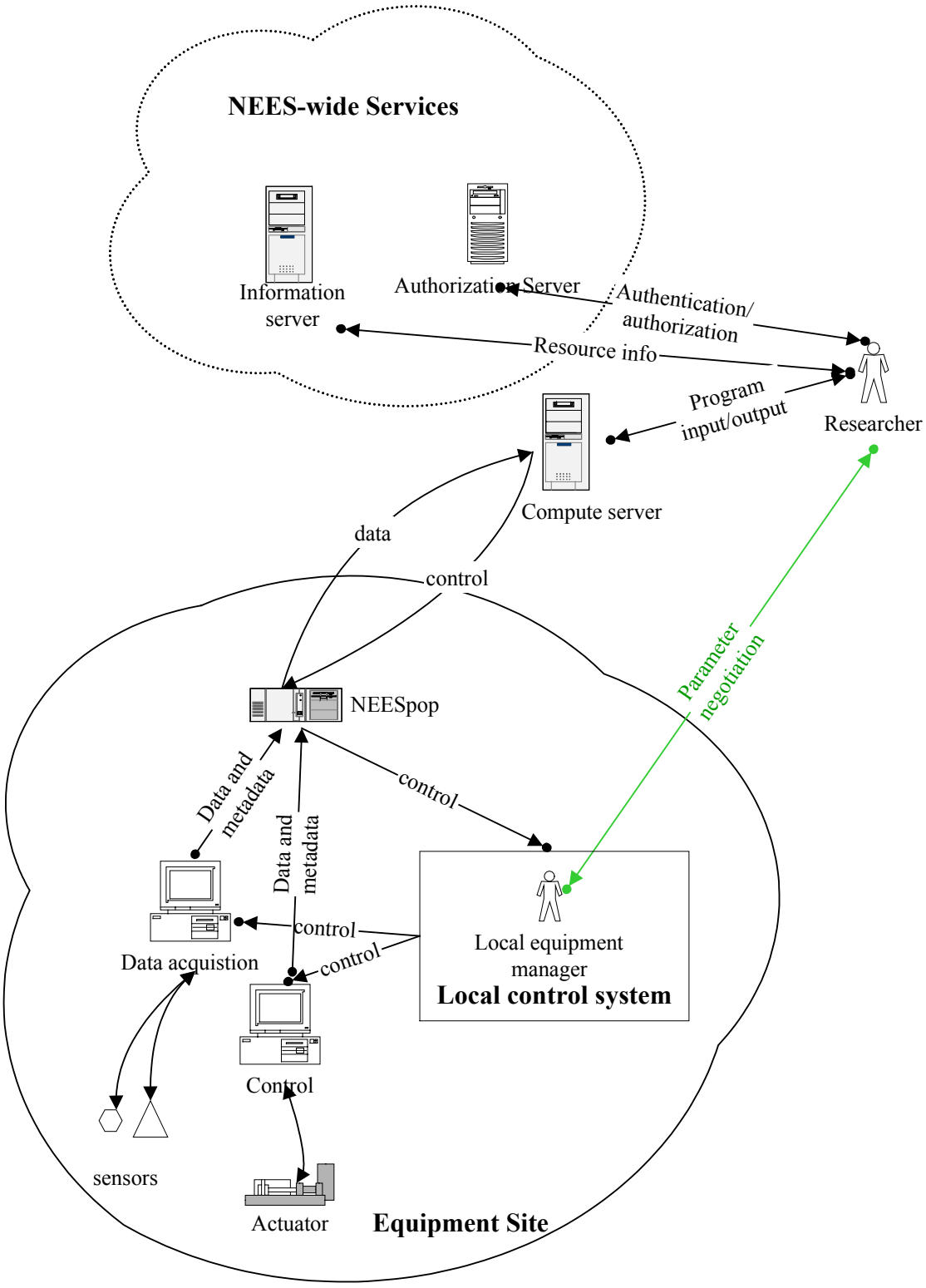

Figure 20: A pseudo-dynamic experiment

Researchers will use essentially the same mechanisms in this scenario as in the previous one. They will make sure that the numeric simulation can accept input using the NSDS protocol and (in this case) send control requests using the NTCP protocol, and they will use the same standard grid components to locate resources and initiate the numeric simulation and the same processes as before to enter descriptive data into the repository, initiate the physical experiment trial, publish results from the physical and numeric simulations, and perform mutual authentication with each server (interactions with the data repository are the same as in previous scenarios and are not pictured in Figure 20). As the trial runs, the numeric simulation (or its underlying framework) will subscribe to the same data streams on the NSDS server on the equipment site's NEES-POP for input and will send additional NTCP requests to control each time-step of the physical trial.

The major difference between this scenario and the preceding one is that the equipment site must have a procedure in place, as described in Section 5.4.2, for accepting NTCP requests to control an experiment. Moreover, the site equipment manager (or other designated individual) must be willing to authorize the researcher to send control requests as well as read experiment data.

7.5 Hybrid Pseudo-Dynamic Experiment

A hybrid pseudo-dynamic experiment combines simulations of several components of a structure. A numeric simulation combines the results of numeric or physical simulations of each component at each time-step to determine the input for each component at the next time-step. In this case, it may be desirable to have several sites run simulations of different components.

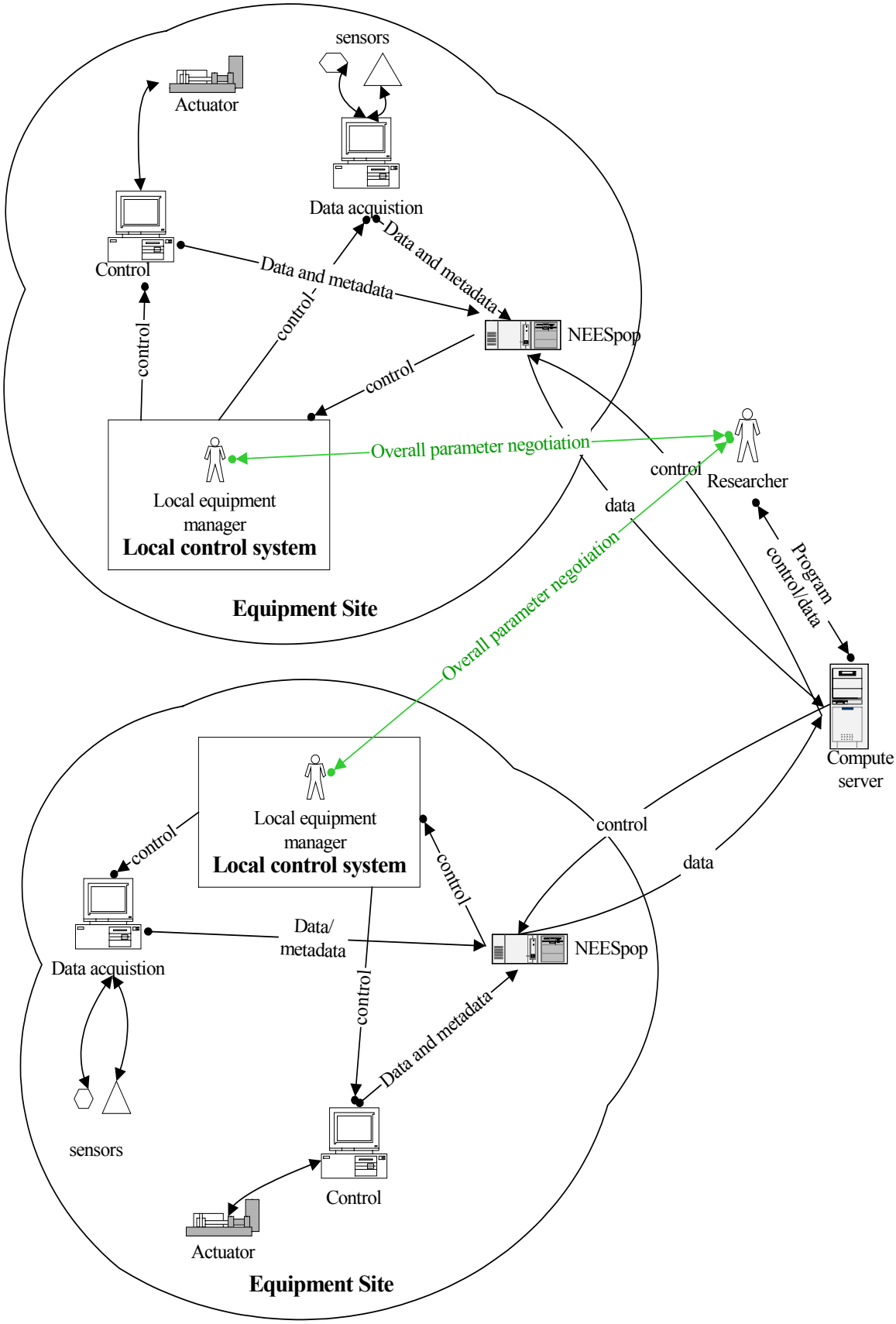
sensors
Actuator
Data acquistion
Control
Data and metadata
Data and metadata
control
control
control
NEESpop
Overall parameter negotiation
Local equipment manager
Local control system
Equipment Site
control
Researcher
data
Program control/data
Overall parameter negotiation
Compute server
Local equipment manager
Local control system
control
control
control
data
Data acquistion
Data/ metadata
NEESpop
control
Data and metadata
sensors
Control
Actuator
Equipment Site

Figure 21: Multisite hybrid experiment

In this scenario, the implementation at each site and the interactions between the researcher and the centrally managed NEES services (not pictured in Figure 21) are identical to that of the preceding scenario. As long as each site uses the NSDS and NTCP protocols for data input and control, any combination of simulation methods can be used at the various sites. For example, some of the participating sites might perform physical simulations, each using different control hardware and software, while other sites might run numeric simulations.

Acknowledgments

This work was supported by the NSF NEESgrid project. We thank the members of the earthquake engineering community for providing valuable input and insight that made the development of this document possible. The following is only a partial list of scientists to whom we acknowledge our gratitude.

University of Reno:

- Ian Buckle
- Manos Maragakis
- Patrick Laplace
- Gokhan Pekcan
- Sheriff Elfass

Oregon State University

- Solomon Yim
- Charles Sollitt
- Cherri Pancake
- Terry Dibble
- David Standley
- Sally Haerer
- Ken Ferschweiler
- Tim Holt
- David Crowe

University of California, Berkeley:

- Jack P. Moehle
- John Canny
- Stephen A. Mahin
- Khalid Mosalam

- Bozidar Stojadinovic
- Don Clyde
- Silvia Mazzoni
- Gilberto Mosqueda

University of Minnesota

- Drew Daugherty
- Doug Ernie

Rensselaer Polytechnic Institute

- Tarek Abdoun
- Christophe DuPre
- Mourad Zeghal

University at Buffalo (SUNY)

- Michel Bruneau
- Andrei Reinhorn
- Mark Pitman
- Kurt Winter

University of California, San Diego

- Fran Berman
- Lelli Hose
- Frieder Seible
- Chris Latham

Bibliography