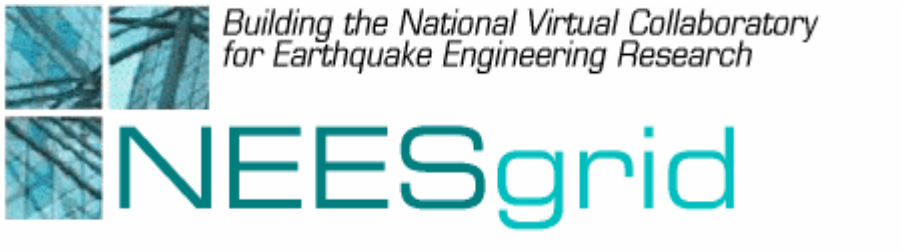

Technical Report NEESgrid-2002-04
www.neesgrid.org

Whitepaper Version: 1.0.2
Last modified: July 2, 2002

The NEESgrid Data and Metadata Harvesting Protocol: an Overview

Joe Futrelle[1] Jeff Gaynor[1]

[1] National Center for Supercomputing Applications, Urbana-Champaign, IL 61820

Feedback on this document should be directed to futrelle@ncsa.uiuc.edu

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.

The NEESgrid Data and Metadata Harvesting Protocol: an Overview

Draft whitepaper by Joe Futrelle and Jeff Gaynor. Version: 1.0.2. Last modified: July 2, 2002

The NEESgrid Data and Metadata Harvesting service will provide NEES users with the ability to transfer data and metadata to the central repository. It will also enable users to manage the data and metadata in a variety of ways, including organizing data and metadata objects into projects, updating and removing data and metadata objects, and controlling who has access to the objects.

Architecture

The NEESgrid Data and Metadata Harvesting service will consist primarily of two software components: a central *harvester* component and a *storage proxy* component for each site. The harvester component will run on the central repository server at NCSA, and the storage proxy component will run on the NEES-POP at each site. Users will control the harvester using a web-based CHEF interface which will allow them to schedule data transfers and browse and manage data and metadata. The harvester will acquire data and metadata from sites by "pulling" data and metadata files from local storage. Users will be able to request that data or metadata files be transferred immediately, at a pre-arranged time, or have the harvester periodically poll local storage to check for new or modified files.

The harvester will provide data and metadata to the indexing and archiving components of the repository, enabling the repository's searching, browsing, and access capabilities. In addition to data and metadata, the harvester will also keep track of *system-level information* for managing the data and metadata objects, such as file type, creator, revision history and authorization. This information will be automatically generated, modified, and used by the data management system in the course of managing data and metadata objects.

The harvester's access to site data will be coordinated by each site's storage proxy component, which will provide the harvester with access to heterogeneous local storage systems at the sites. The storage proxy will coordinate this access using one of the following supported local mechanisms:

- *GridFTP access to the local storage server*. The harvester transfers data and metadata directly from local storage using the GridFTP protocol. The site's storage proxy component coordinates this access.
- *Local disk on the NEES-POP*. Sufficient disk space is added to the NEES-POP to store data and metadata. The site's storage proxy component provides the harvester with GridFTP access to this disk space.
- *File system NFS-mounted on the NEES-POP*. A directory on a file server on the site's LAN is mounted on the NEES-POP using NFS. The site's storage proxy

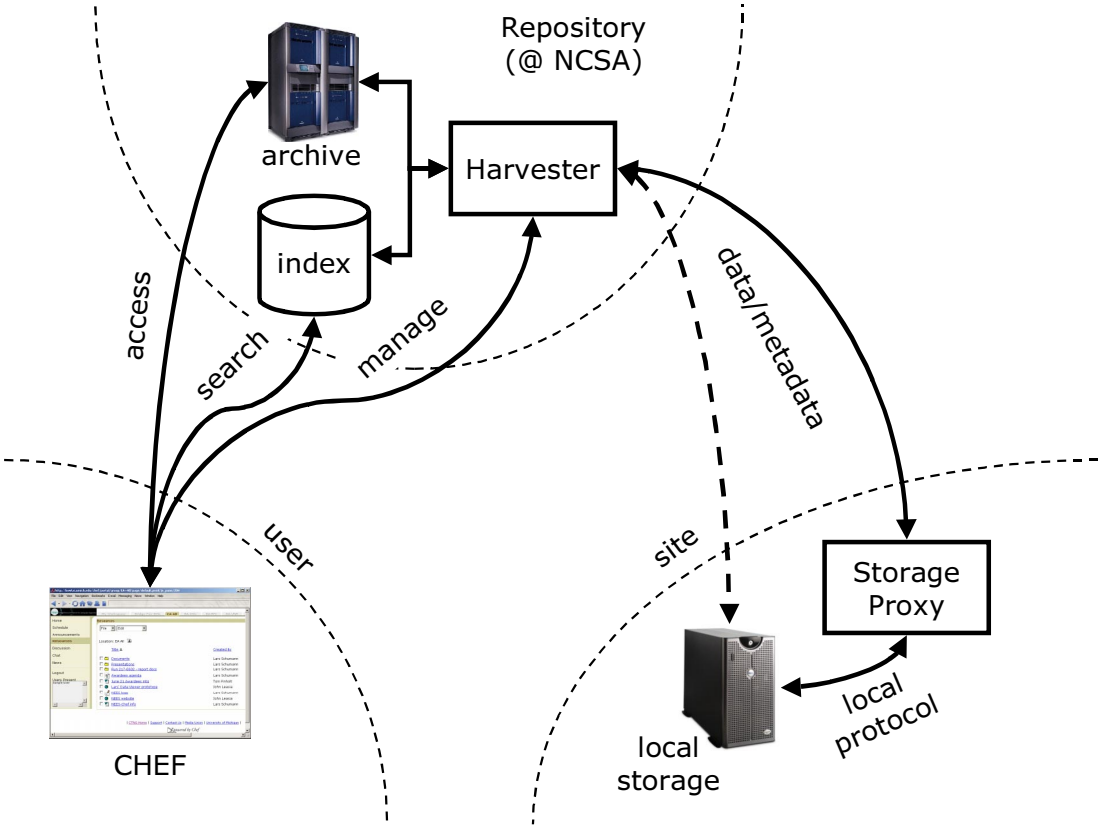

Figure 1: The NEESgrid Data and Metadata Harvesting Architecture.

component provides the harvester with GridFTP access to this directory. For this mechanism to perform adequately, the LAN connection to the file server must have gigabit throughput.

The CHEF interface will provide a web-based means for researchers to control the harvester in order to manage the movement and organization of data and metadata. Users will authenticate through the CHEF interface, providing them with secure access to the harvester's functionality. They will be able to manage data in the following ways:

- Schedule transfer of data and metadata objects from site to repository.
- Add/delete/update data and metadata objects in the repository.
- Browse hierarchies of data and metadata objects and organize data into projects
- Modify access permissions associated with data and metadata objects

The harvesting architecture is shown in Figure 1. Note that because CHEF is web-based, the user may be located at the site or elsewhere; control over the harvester's actions will thereby be able to be distributed among remote collaborators.

Object Model for System-level Information

The repository and harvester will organize data and metadata objects according to a simple object model allowing data and metadata objects to be described and managed.

The object model will include object hierarchies such as projects, experiments, and tests, as well as object types such as files, users and servers. Other system-level information associated with each object will include the object's file type, creator, revision history, and authorization parameters.

The system-level object model will be mostly independent of the NEES comprehensive data and metadata model, which serves to describe equipment, experiments and tests as completely as possible. Instead, the system-level object model will be designed to allow data and metadata objects such as files to be described and managed. It will serve as the means of describing where NEES data and metadata objects are located in the repository, how they are encoded, and who has access to them.

The model will allow unique *URI's* (Unique Resource Indicators) to be associated with NEES data and metadata files, and thus the objects contained in them. Each data and metadata file will have a URI which can be used as a base URI for objects contained in it. For instance, a STEP file might contain hundreds of objects, each with its own STEP-specific identifier. URI's could be constructed for each of these objects by appending the STEP-specific identifier to the end of the STEP file's URI. This scheme will allow objects to cross-reference each other so that complex relationships will be able to be described between objects even if they're contained in unsupported or partially-supported formats.

Security Model

Security for harvesting and data management will be implemented using a combination of Grid-based authentication and NEES-specific authorization systems. A directory of security entities such as users and servers will be managed centrally which will associate entities with Grid certificates. Users entering the system through CHEF will acquire Grid certificates either directly or using the CHEF service as a proxy. Once authenticated, users will be authorized to manage and use data and metadata objects, and to transfer site data to the repository.

Access rights will be determined on a per-file or per-object basis, depending on the action that a user wishes to take. There will be a hierarchy of access levels associated with users and groups will prevent users from acquiring unauthorized access to data, ranging from administrator access to read-only access. Access rights will also be able to be applied to project groups or sites, so that each user in that project group or site will be given a certain level of access to the relevant objects. For example, all users at a certain site may be permitted to access information about the site's experimental apparatus, or all users in a particular collaborative team may be given exclusive access to work with data produced during their ongoing collaboration.

The kinds of access that will be able to be allowed or disallowed include:

- browsing existing data and metadata
- adding/deleting/updating data and metadata
- modifying group membership
- modifying data/metadata object access rights
- searching
- downloading/viewing data and metadata

Glossary

Authentication. The process by which a security system determines and validates the identity of a user or other security entity.

Authorization. The process by which a security system determines the access rights associated with a user or other security entity.

Certificate. A digital security credential issued as part of an authentication process.

CHEF. The web-based collaborative environment which will provide users with access to NEES resources.

GridFTP. A Grid-enabled version of the File Transfer Protocol, which provides performance, security, and reliability enhancements.

Harvester. A software component which gathers data and/or metadata from distributed sources.

Identifier. A label that serves to distinguish one object from another.

Metadata. Data which describes or otherwise contextualizes other data.

NEES-POP. The NEES "Point Of Presence": a server located at each NEES site that provides the site with Grid and NEES services. See the NEES-POP whitepaper for more details.

NFS. Network File System. A protocol which provides local mechanisms for accessing distributed file systems.

Object model. A conceptual framework which decomposes problems into objects which have properties and behavior associated with them. This is the basis of object-oriented programming, but can be used for data modeling as well.

Security entity. An entity (such as a user, software component, or hardware component) which can authenticate to a security system and be authorized to perform various actions.

STEP. Standard for the Exchange of Product Model Data (ISO 10303). STEP is a comprehensive, widely-used standard focusing on representing the geometry of complex physical objects.

Storage proxy. A software component which provides other components (such as harvesters) with access to storage resources.

System-level information. Information about data objects that allows a data management system to manage the objects.

URI. Uniform Resource Indicator. A standard means for identifying digital objects in networked environments. URL's (Uniform Resource Locators) are a specialization of URI's.