

CASFI  
Collect, Analyze, and Share for the Future Internet



# CASFI Data Sharing Platform

Nicolas Aycirieux nicolas.aycirieux@gmail.com  
Sue Moon sbmoon@kaist.edu

from **KAIST**



This work was supported by the IT R&D program of MKE/KEIT [KI001878, "CASFI: High-Precision Measurement and Analysis Research"]

## Abstract

Measurement-based research benefits greatly from efficient and minimum-effort management of large-scale data and their metadata. Effective data management has become a key in promoting data sharing within our community. During the past decade, a great deal of effort has been devoted to building Internet data archives. Several institutions run repositories where they post metadata of their collections, along with the data themselves, if their size is not prohibitively large. However, these repositories have been developed independently with different user interfaces, which adds complexity when users look for data of relevance scattered all over the Internet.

CASFI (Collect, Analyze, and Share for Future Internet) is a team of four laboratories that collaborate actively in measurement-driven research. In order to facilitate our collaboration, we propose a new data management system called the Data Sharing Platform (DSP) that can be used by any institution to manage its own data and share it easily. The DSP is flexible in order to adapt to any infrastructure, and assists users with the data registration process. Our platform not only manages local data but also integrates remote data in its local database, thus offering a consistent user interface to browse and search the local data archive as well as other remote archives.

Our system standardizes the entire process from data registration, uploading, browsing, and search. It is implemented and deployed at our four institutions and tested extensively. We hope to expand its user base to greater research community.

**CASFI Data Sharing Platform**

## Overview of the existing solutions

	<ul style="list-style-type: none"> <li>▪ Personal depository of Internet traffic data</li> <li>▪ Can be browsed on the Internet</li> <li>▪ Data are not downloadable</li> </ul>
	<ul style="list-style-type: none"> <li>▪ Daily traces of the WIDE backbone since 1999</li> <li>▪ Scripts to automate the data registration</li> <li>▪ Most of the traces are downloadable</li> </ul>
	<ul style="list-style-type: none"> <li>▪ Archives data from wireless networks</li> <li>▪ Publishes tools used in data collection and analysis</li> <li>▪ Contribution is encouraged (manual integration)</li> </ul>
	<ul style="list-style-type: none"> <li>▪ The most community-oriented metadata depository</li> <li>▪ A tool is provided to contributors to register their data more easily (automatic metadata extraction)</li> <li>▪ Great communication effort</li> </ul>

**OPENNESS TO THE COMMUNITY**

**2 / 13** **NOMS 2010 – 19~23 April**

In this slide we give a brief introduction of four repositories with the most extensive sets of data. They vary in what data is available and who can contribute.

The WAND network research group at the University of Waikato manages metadata of Internet traffic data they own through WITS (Waikato Internet Traffic Storage). Their collections, which are mostly packet-level traffic captures on high-speed Internet links, can be browsed via the web interface but the data cannot be downloaded.

The WIDE (Widely Integrated Distributed Environment) group has a similar archive, called MAWI (Measurement and Analysis on the Wide Internet). Their traffic archive has daily traces of the WIDE backbone (several sample points) since 1999. The registration process of new traces is automated. In addition to the features found in WITS, most of the data are publically available.

CRAWDAD (A Community Resource for Archiving Wireless Data At Dartmouth) archives data from wireless networks: mostly from 802.11 wireless LANs, sensor networks, and MANETs. Their data is available for download and they also publish tools used in data collection and analysis. They encourage researchers to share they wireless traces on CRAWDAD, but the integration is manually done by the CRAWDAD team.

DatCat (Internet Measurement Data Catalog) of CAIDA (Cooperative Association for Internet Data Analysis) purports to be the repository of the metadata of the entire network research community; in this sense the most community-oriented. No data is available on the website, but only the metadata. The creators have made a great and continuous effort to encourage researchers to register their metadata. Once registered, users can contribute in a more structured approach than with CRAWDAD since a metadata generating tool is used to extract necessary metadata in the XML format. This way all the necessary metadata is automatically catalogued.

These repositories (and many others) have been designed according to requirements and goals specific to host institutions. As a consequence they have different interfaces and functionalities, which present complexities to end-users.

## DSP's design goals

A data management system **common** to every institution would be a step towards **standardization** of measurement data description and easy **collaboration**.

### ➔ Development of the Data Sharing Platform

- installed in every institution
- manages local data and metadata
- integrates other institutions' information in the local database

### Design goals

#### Flexibility

- adaptability to any infrastructure
- management of any type of data (well-known formats as well as custom formats)

#### Simplicity

- automatic metadata extraction
- assisted custom data type creation
- batch registration of files having the same characteristics

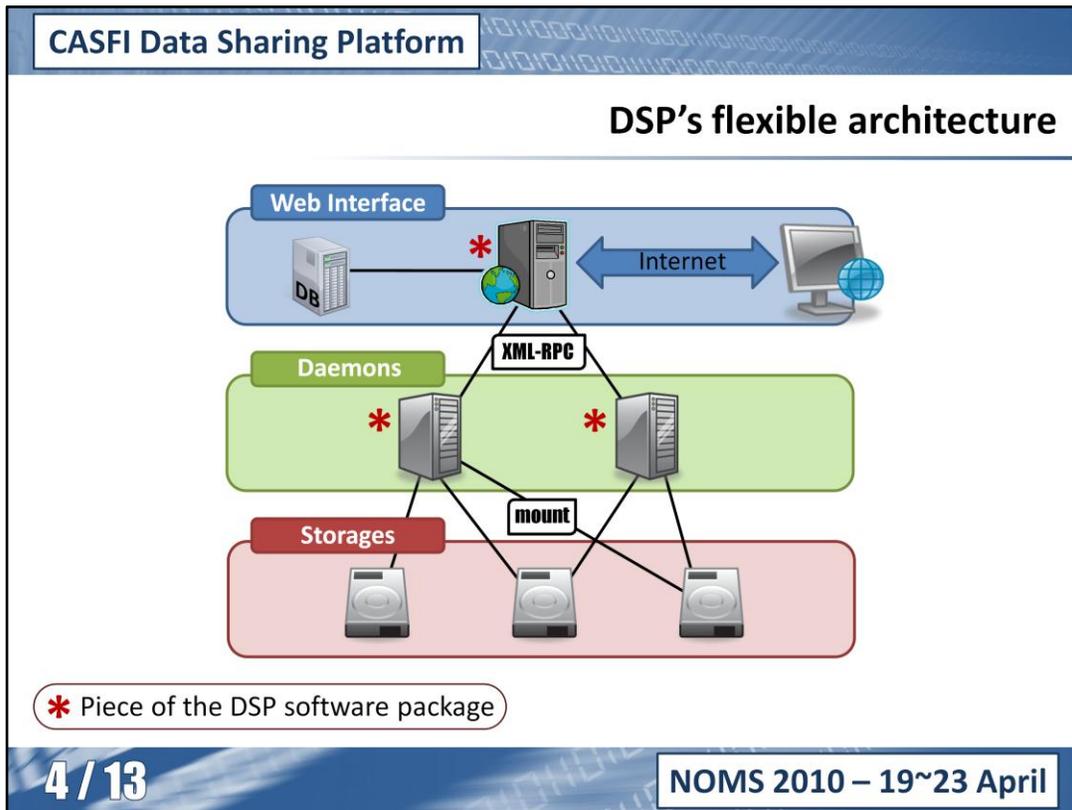
For those institutions with data archives, the process of data collection, analysis, and sharing can be time consuming and complicated, involving different people and tools at different phases. Our objective is to provide a data management platform that can be used by any institution. Our Data Sharing Platform manages local data and metadata, and automatically integrates other institutions' information as well into the local database, turning each DSP into a global repository. Our design goals are:

#### Flexibility

- Each institution has different storage systems, and DSP must support a wide range of system configurations. Our DSP platform can run on a single machine, and easily scales up as the system grows.
- Our community uses a broad range of data, from traces of a well-known type (e.g. pcap) to custom types (e.g. web-crawled text data). DSP supports all types of data.

#### Simplicity

- DSP offers assistance to users during data registration. Some key fields in metadata are automatically extracted from the submitted files.
- When a custom data type (a new type that is not known to the system) is created, a few lines from the file are extracted as sample and an on-the-fly generated form is presented to the user for ease of input.
- Files with the same characteristics can be registered in a batch, saving a substantial amount of time.



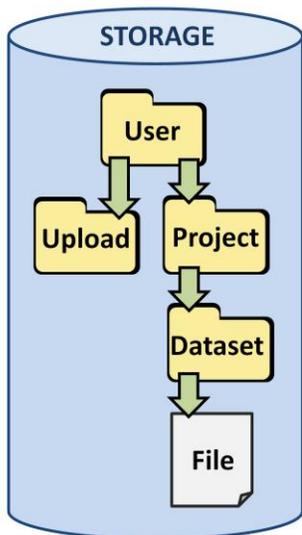
Our system has a three-layer architecture.

The web interface layer is the front-end hiding the complexity of DSP. It contains the web server that handles all the requests from and the responses to users, and a database management system that stores the metadata. Any DBMS can be used.

The daemon layer contains one or more machines whose role is to bridge the web interface layer and the storage layer. These daemons listen for XML-RPC requests from the web server, and execute read and write to storage. Multiple daemons are a convenient feature at a site where storage systems are segmented and are not accessible globally. They also offer reliability in case of a daemon machine failure. For future work we plan to add a load balancing mechanism between daemons.

The storage layer contains all the storages that are used by DSP to archive data. Any type of file system can be used, as long as the daemons can be configured to access the data. Simply put, a storage refers to a file system partition that can be mounted.

The web interface, the daemon, and the storage can all be on one machine or spread over multiple machines. The layering architecture offers each site flexibility to configure DSP best suited to the hosting site's own needs and resources.



- **Project:**

The main theme of research.

*Ex: KAIST Traffic*

- **Dataset:**

Group of files that have the same characteristics (same purpose, same creation process, same format).

*Ex: Backbone link traffic, sample point B*

- **File:**

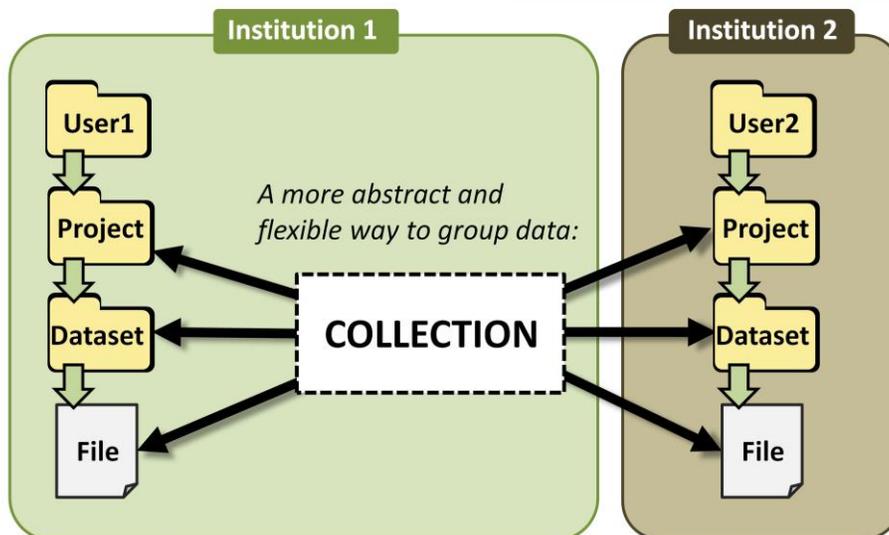
Any binary or text format.

*Ex: hourly erf traces*

In DSP data is organized hierarchically. At the top there are users. A user can have multiple projects. Each project in turn contains one or more datasets (e.g., datasets collected on different dates for the same project or heterogeneous types of data from the same project). Files that belong to the same dataset must have the same format. The hierarchical data organization is to help users to keep their projects and related data files together in a coherent manner. It also helps other users to browse the data later and understand the inter-dependency.

Files belong to the user who deposits them to the system. What if another user would like to use it? In order to address file sharing between users, we introduce the concept of a collection in the next slide.

## Data organization (2)



A user owns the entire hierarchy of data, but often users share data in their research. For example, a researcher could work with data that belongs to one's colleagues. Or multiple researchers from different laboratories could collaborate on a common project. In order to support diverse forms of collaboration DSP presents the concept of a collection. A collection is a set of symbolic links (in the database) pointing to data of one's own (projects, datasets, or files), colleagues' data, or even other institutions' data.

When a researcher writes a paper, all the data used in the work can be grouped as a single collection. Anyone can access all the necessary information relevant to the work via the collection.

**DSP user interface**

- developed with the Django web framework (Python), and jQuery (JavaScript) to add interactivity and responsiveness
- functionalities:
  - ✓ submit data
  - ✓ browse data and collections (different access levels)
  - ✓ search
  - ✓ administration (users, daemons, storages management)

➔ <http://casfi-dsp.kaist.ac.kr>

**Global website**

- the institutions can register and download the DSP package as a customized tarball
- provides guidelines for installation, configuration, and utilization

➔ <http://download.casfi-dsp.kaist.ac.kr>

**DSP user interface**

The front-end of DSP is implemented with Django, a powerful web framework based on the Python programming language. It follows the Model View Controller architectural pattern that helps with the source code maintenance. We also use jQuery, a JavaScript framework that facilitates the modification a web page dynamically and the emission of asynchronous HTTP requests. It makes our interface interactive and responsive.

After creating an account (an administrator's manual validation required), users can submit new data for registration, browse data and collections of the institution as well as the public ones from other institutions, and search. Visitors without accounts can only see and possibly download the public data of the institution.

**Global website**

Institutions can register and obtain the DSP package on the DSP global website. This package is customized according to the information provided by the institution during the registration process, and then made downloadable as a tarball. Once the package has been downloaded, the person in charge of the DSP at the new institution can refer to the manual page and follow the instructions of the installation, configuration, and utilization guides.

**Main daemonized program**

- listens forever to the XML-RPC requests coming from the web server
- simple file system operations (read, move, delete data)
- more complex functionalities (checking data integrity, handling file downloading by chunks, etc.)

**Metadata extraction tools**

- standalone programs used to extract metadata from a given file
- 3 tools bundled into the DSP package: timestamp extraction for pcap format , erf format, and custom text format

**XML Configuration file**

- describes IP and port where to listen for incoming request, IP of the web server (filter)
- mapping between the data types described in the database and the metadata extracting tools related to it
- easy to hook a new metadata extraction tool by adding a line in the file

The daemon module, whose role is to handle the interactions between the web interface and the storage, is composed of:

**Main daemon program**

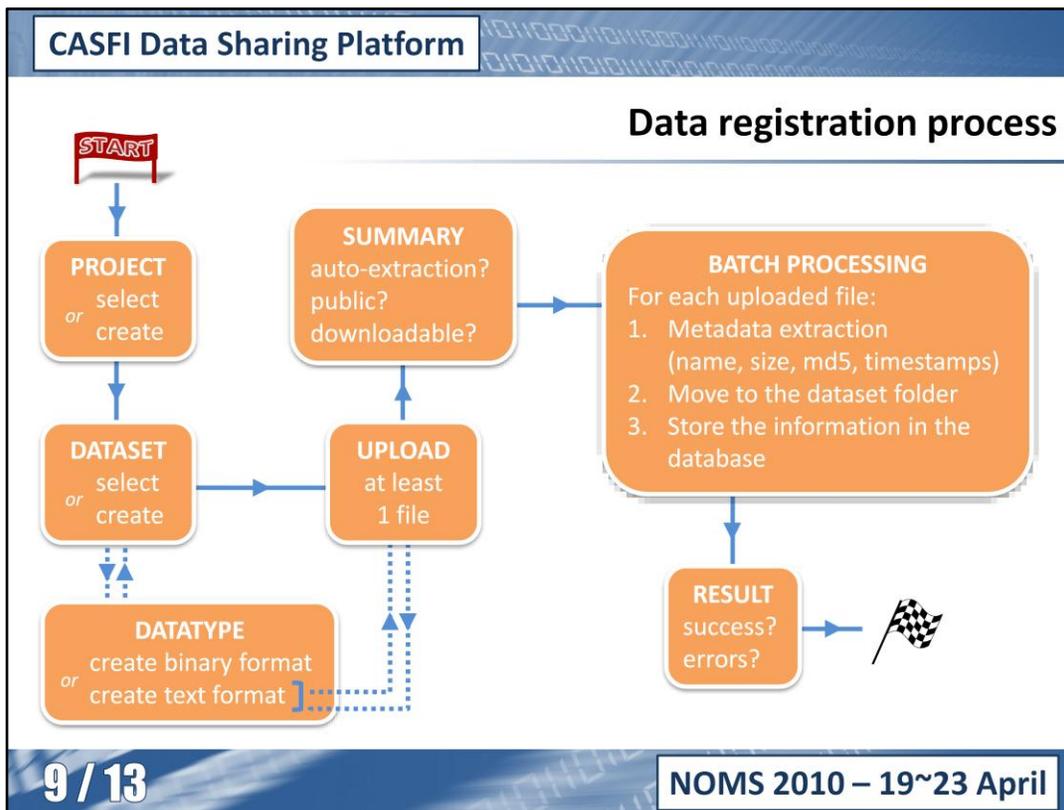
The main program implements the set of functions that access the storage with the parameters set by the user via the web interface. Most of them perform basic file system operations, such as reading, moving, or deleting data. Other more complex functions are checking data integrity or handling file downloading by chunks. The daemon loops forever, listening to the XML-RPC requests coming from the web server.

**Metadata extraction tools**

A *tools* folder contains the tools that the main daemon program uses to extract metadata from a given file. Three short programs that extract start and end timestamps are bundled into the DSP package: one for pcap format traces, another for erf traces, and the last used as a default program for free-format text data. Other institutions are encouraged to develop their own extracting tools for their custom data types. The current version of our DSP extracts only timestamp-related metadata, and could be expanded to extract more fields.

**Configuration file**

The configuration of the XML-RPC server is stored as an XML file. The administrator chooses the IP address and the port of the daemon and the IP address of the web server. The configuration file also provides a mapping between the data types described in the database and the matching metadata extracting tools. When a new tool is developed, its absolute path must be written as a new XML element in the configuration file, and this element's tag name must match the name of the data type in the database.

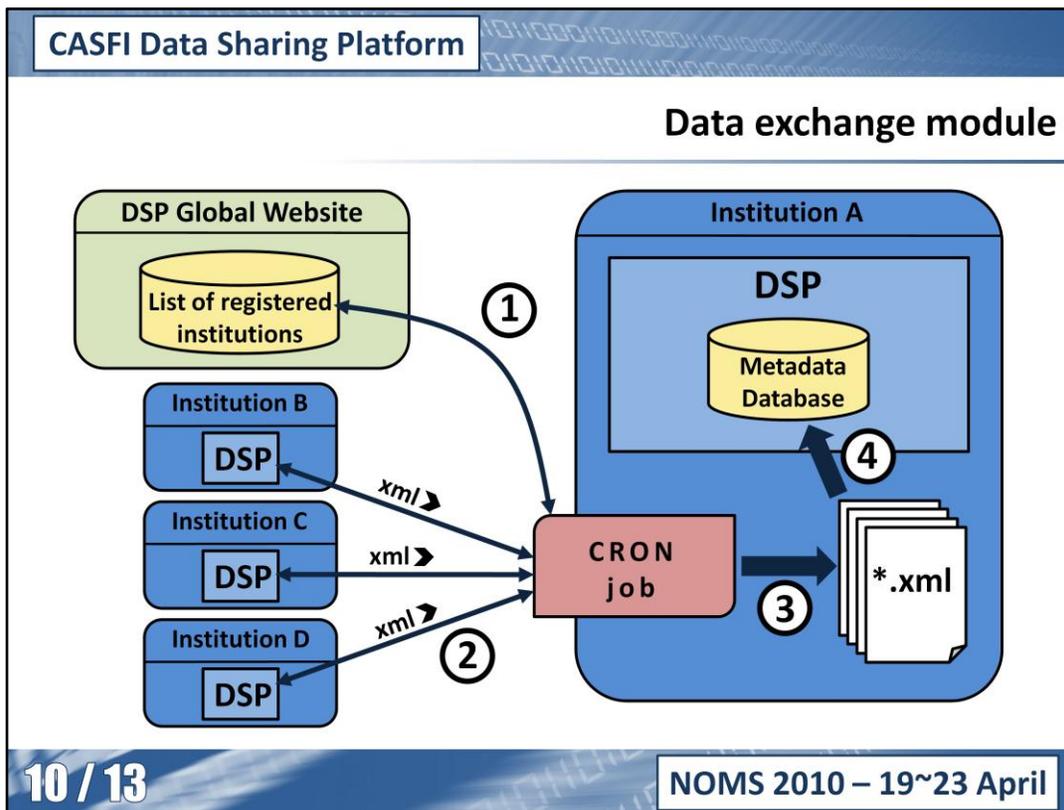


The first step in data registration is to specify the location of data: a user must select an existing project or create a new one (defining its label, folder name, storage, and description), and then select a dataset or create a new one (defining its label, folder name, data type, time zone, and creation process). A custom data type can be created during the dataset creation. If it is in binary format, the user is asked to describe the structure manually. If it contains text, the user is asked to upload the files first in order to be able to extract a sample data row, split the columns into tokens, and generate a form tentatively filled with those tokens (semi-automated data type creation).

Once the project and the dataset have been defined, the files must be uploaded to the user's upload folder, on the right storage. This storage might be handled by one or several daemon servers, and the list of these daemons (IP address, alias) is displayed along with the absolute path of the upload folder (built from the path where the storage is mounted concatenated to the path of the DSP folder on this storage). Thus, the user can upload files without access to the storages, through one of the daemons (the most conveniently reachable one). There is no restriction regarding the transfer method: currently FTP, SCP, and SFTP are offered. The system handles compressed and uncompressed files transparently.

Provided that at least one file has been uploaded, a summary with all the previously specified information is displayed. If a metadata extracting tool is available for the data type, the user can decide whether to use it or not.

The files are processed one by one. For each file, the mandatory name, size, md5 and the optional timestamps are extracted, the file is moved to its dataset folder, and the metadata is stored in the database. These three steps are considered as a single transaction, and if an error occurs, the entire transaction is undone. Finally, the result of the registration is displayed.



DSP keeps metadata from other institutions locally up to date by periodically fetching those metadata and integrating them to local database. Local and external data can be browsed, searched, and downloaded through the same consistent interface. This process is implemented as a cron job and has four steps:

#### 1/ List of institutions

The global website stores the list of registered institutions. The script knows the URLs to access in order to get all the necessary information about those institutions in an easily parseable text format (one line for one institution). In case the global website is down, the previously stored list is used. If one of the DSPs does not respond, it is ignored after a timeout.

#### 2/ Fetching the data

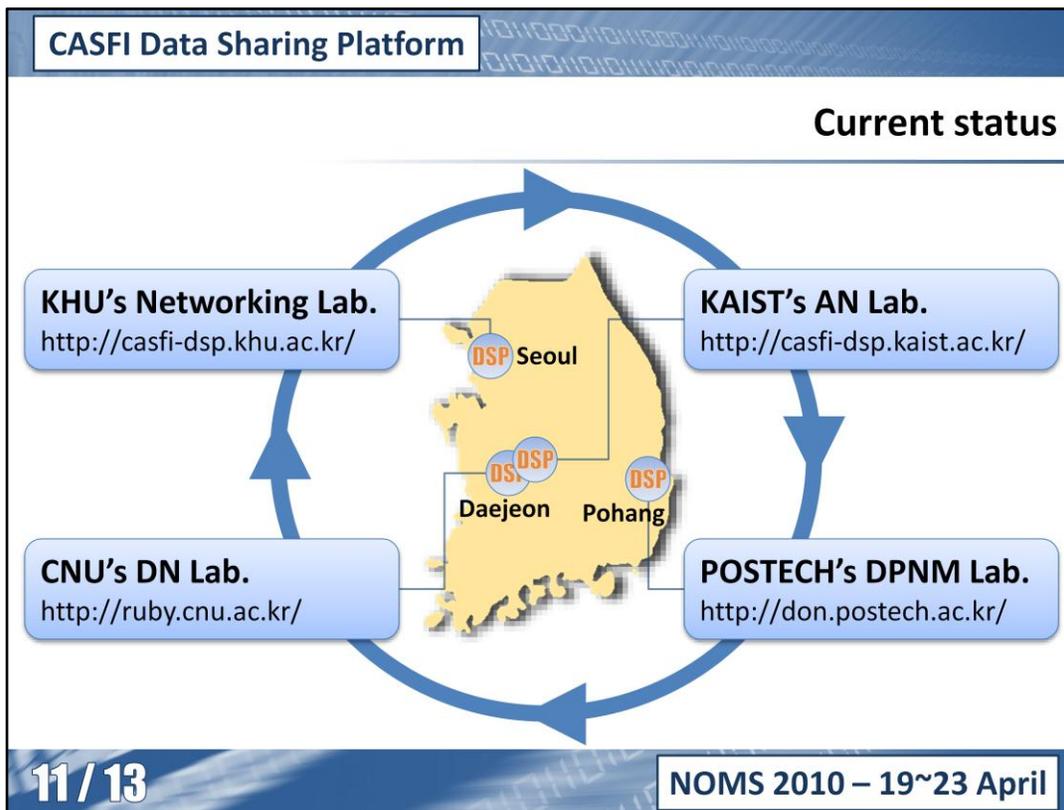
For each institution in the list (B, C, and D), A invokes their *export* function. An XML document containing all of their public data is generated.

#### 3/ Storing the documents

When the XML document is generated, some fields are left blank because the required information is not in their database, but in the global website's database (for example, the URL of their DSP or the email address of their administrator). These fields are filled by the script and the completed XML documents are written in a specific folder.

#### 4/ Integrating the data

Finally, the script invokes the local *import* function that parses each XML document and integrates the contained data in the local database.



First, the data management module was made available to the members of the Advanced Networking Laboratory (KAIST). They registered different types of data, standard format traffic traces as well as free-form text files. Based on the feedback we fixed major bugs and added several functionalities. Once the data exchange module was integrated to the DSP, a beta version was released on the global website. We helped the Data Networks Laboratory (ChungNam University) to install and upload data. In particular we could test the metadata exchange process between two institutions. Finally, the DSP was installed in the two other remaining teams of CASFI, the Distributed Processing and Network Management Laboratory (POSTECH) and the Networking Laboratory (KyungHee University). The administrators of these laboratories managed to install the system by themselves, following the instructions of the completed manual.

All the members of CASFI are now running a stable version of the DSP.

## Future work: remote processing

### Constraints for sharing data

- privacy issues (traces containing privacy-threatening information such as IP addresses)
- files can be prohibitively large

### Solution: remote data analysis

- fine-grained control over the set of programs that can be run safely
- the outputs are stored along with the file and are visible by anyone

### Challenges

- a wide range of programs could be used
- share the processing power fairly among users having different levels of priority

Current DSP does not allow any data processing via a query. Following issues remain in order to enable query-triggered processing on DSP.

First, we should introduce the multiple level of a file's visibility. Who can access what part of the file? For example, IP addresses of a packet could identify a user and one's activities, while the size of a packet is less informative. Or a summary instead of the full data file. If we could support query-driven data processing.

The concept of remote processing is pretty straightforward but challenges arise. Any analysis program could be used. This implies that the DSP must be extremely flexible regarding its way to hook these programs and to handle any sort of output. We plan to handle simple programs that can output only text data to begin with, and add tools gradually. Another issue to address is the management of processing resources. Since several users with different levels of authority can execute jobs simultaneously, we must manage the resources properly in order to share the processing power fairly, and according to the priority of the task.

**The CASFI Data Sharing Platform:**

- is **common** to all the institutions of the measurement-based research community
- is **flexible** and **simple** in order to suit as many institutions as possible
- offers **three features**
  - ✓ efficient local data and metadata management
  - ✓ automatic fetching and integration of external metadata
  - ✓ remote data analysis (not implemented yet)

BUT the biggest challenge remains: convincing the community to adopt it  
Our advantage: the DSP is already used by 4 laboratories



at <http://download.casfi-dsp.kaist.ac.kr>

We have described the CASFI Data Sharing Platform, our data management system. Its goal is to serve the measurement-based research community by facilitating data and metadata sharing and encouraging collaboration. Flexibility and simplicity are the two key design goals.

The main features of the DSP are efficient local data and metadata management, automatic fetching and integration of external metadata, and remote data analysis (not implemented yet). Remaining technical issues aside, the big challenge is to convince the community of its value and encourage them towards a unified solution. Our advantage is that the DSP is already up and running in four laboratories so that potential users can see concretely the benefits of such a system, and hopefully join us in order to participate the construction of a global Internet data depository.

**References**

1. CASFI. <http://casfi.kaist.ac.kr/>
2. WITS. <http://www.wand.net.nz/wits/>
3. MAWI. <http://mawi.wide.ad.jp/>
4. CRAWDAD. <http://crawdad.cs.dartmouth.edu/>
5. DatCat. <http://www.datcat.org/>
6. Mark Allman, Ethan Blanton, and Wesley M. Eddy. A scalable system for sharing Internet measurement. In In Proceedings of Passive & Active Measurement (PAM, pages 189-191, 2002.
7. Sue Moon and T. Roscoe. Metadata management of terabyte datasets from an IP backbone network: Experience and challenges. In ACM SIGMOD Workshop on Network-Related Data Management, Santa Barbara, CA, May 2001.
8. Douglas C. Sicker, Paul Ohm, and Dirk Grunwald. Legal issues surrounding monitoring during network research. In Proceedings of the 7th ACM SIGCOMM Internet Measurement Conference (IMC), pages 141-148, New York, NY, USA, 2007. ACM.