DATA SHARING III: TOOLS AND TECHNIQUES TO ADVANCE INTEROPERABILITY

Adrian Robert, Michael Abato, and Daniel Gardner
Laboratory of Neuroinformatics, Dept of Physiology & Biophysics, Weill Cornell Medical College, NY, NY

INTERNALS OF NEUROINFORMATIC RESOURCES
In a parallel paper (Kastner et al, Data Sharing II, these meetings), we offered standards for interface protocols and representations between neuroinformatic resources. In this paper, we present a complementary view of the internal—models, architectures, and development strategies—of such neuroinformatic resources.

The Human Brain Project Program Announcement (PAA-03-015) calls for "extensible, scalable and interoperable" neuroinformatic resources to... present a plan for continued updating and maintenance. Toward these goals, we propose design choices to aid development and implementation of efficient, maintainable, persistent, and interoperable models to support data sharing. The rapid growth of networked enterprise management and customer service applications in the commercial world has produced a rich set of commodity server software. Many of these, including database management systems, languages and application development platforms for scalable deployment, and methods for sharing source code—can be utilized for neuroinformatics projects. We provide an overview of these major technologies, distilling pertinent characteristics and utilization considerations.

MODELS FOR DATA STORAGE
Open Source Databases

Open databases are today more available than ever. Commercial vendors, of course, manage the same business models as in previous years. But now open source 'shareware' is more available, and the open source movement enjoys considerable 'maturity' compared with a few years ago. Considered second-tier by many, but 'open source' now is, in many cases, the best and/or most cost-effective choice. The most popular open-source database is MySQL, the Swedish shareware of the year 2000. A brief mention also is given to PostgreSQL, an older, but more mature, commercial product.

1. Open Source vs. Commercial Databases
When does a project need a commercial database? To serve a large organization (particularly in the Java world) implementation (choice of vendor, languages, etc.) may be varied across these boundaries.

2. Alternatives to Relational Databases
When should a project consider alternatives to a relational database? This depends on the structure and quantity of data and the types and frequencies of search and retrieval the system will support.

1. Overview of Server Architectures
In terms of server architectures, while interesting grid and peer-to-peer technologies are developing, the standard client-server model is fully mature. A three-tier server architecture is typical, in which a "logical" layer mediates between the data storage and/or computational components and the application seen by the user.

2. Grid or Agent Framework
Architecture characteristic of grid and agent-based frameworks Clusters (which may also be running in a client-server mode) itself) requests initially to a central broker. The broker analyses the request and specifies a node to satisfy it. It may either continue mediating the communication, or pass the connection information directly to the client.

3. Overview of Server Architectures
In terms of server architectures, while interesting grid and peer-to-peer technologies are developing, the standard client-server model is fully mature. A three-tier server architecture is typical, in which a "logical" layer mediates between the data storage and/or computational components and the application seen by the user.

4. Tiered Client-Server Architecture
Standard three-tier client-server architecture seen in many web applications. Scaling may be handled independently at each layer, by utilizing clustering and related technologies. This approach is used in first-generation consumer file-sharing approaches including Napster and Audio-Galaxy, as well as many current computational grid technologies and software agent platforms.

5. Distributed Brokering Grid
Grids are an architecture in which service information is cached by brokers, which use the information to act as brokers. Avoids scalability problems that can overwhelm a central broker, and may therefore be federated. Again, the benefits are evident when looking at the overall architecture. The problem is now one of the many standard for some things (e.g. inter-node communication), not enough for others (semantic alignment).

3. Overview of Server Architectures
In terms of server architectures, while interesting grid and peer-to-peer technologies are developing, the standard client-server model is fully mature. A three-tier server architecture is typical, in which a "logical" layer mediates between the data storage and/or computational components and the application seen by the user.

7. Client Application Models
Client applications can be categorized into worktation, web-based, and hybrid types. To orthogonal to this is the distinction between applications and APIs (Application Programming Interfaces) or libraries. APIs distribute the burden of developing client applications, and it can be beneficial to publish them even when you are providing applications yourself, to encourage further development and wider adoption. They can be provided at the local workstation level (shared and/or dynamically-loaded libraries), or exposed over CORBA, GUI, or SOAP, etc.

8. Summary of Open Source Licensing Rights and Requirements
Questions to ask to select the appropriate open source license:
1. Can you redistribute the code, including in a commercial product?
2. Must you include the copyright notice with redistributed code?
3. Are licenses to use associated patents included in the licensed code?
4. Must you include the source to the free portion with redistributed or combined versions?
5. Must you make any code added to it in redistributed or combined versions?

9. Summary of Open Source Licensing Rights and Requirements
Questions to ask to select the appropriate open source license:
1. Can you redistribute the code, including in a commercial product?
2. Must you include the copyright notice with redistributed code?
3. Are licenses to use associated patents included in the licensed code?
4. Must you include the source to the free portion with redistributed or combined versions?
5. Must you make any code added to it in redistributed or combined versions?