The Data That You Won't Find In Databases: Tutorial panel on data exchange formats

Peter Buneman David Maier

University of Pennsylvania and
Oregon Graduate Institute of Science & Technology
email: *Peter@cis.upenn.edu, maier@cse.ogi.edu*

1 Focus of Panel

Data exchange formats were originally devised for moving data between programs and between groups of researchers in a platform-independent file format. They are mostly self-describing, containing data element definitions along with the base data, though in some cases they involve a standardized external data dictionary. DXs allow exchange of data structures between programs, not just byte streams. They tend not to support a behavioral component as part of the interchange format, though some have assertions and derived data elements. They are typically implemented as a procedure library that is linked with an application, along with some stand-alone utilities.

An interesting phenomenon is that DXs are being used for data management, although they were originally intended for data exchange. Research groups keep their data in files using one of these formats, and reprogram their tools or write adaptors to use data in that format. The self-description means that, as with a database management system, there is no longer a dependence on a particular application program in order to be able to read and decode the data. This use of DXs for data storage is acknowledged in the tools that are appearing, such as DX-file browsers and cataloging facilities. We also see some DX standards being extended to include new access methods, such as the record-like reads and writes to files present in recent versions of netCDF. Thus, developers know their formats are now being used for data storage and not merely for data exchange.

The question then arises: Why bother with a database management system when DXs are available? There are many advantages to using DXs for data storage. The software for accessing these files is easy to port and has been widely ported—it is often available on high-end and low-end machines where DBMS offerings are scarce. The link libraries exist for languages of interest, for example FORTRAN in the case of scientific DXs. (The number of object-oriented databases with a FORTRAN API can be counted on the fingers of zero hands.) The APIs typically read and write large chunks of data to and from program data structures, which can be more attractive than the cursor- or method-oriented interfaces of many DBMSs. Commercial software is now appearing that can access certain DX files directly, and particular DXs are gaining a measure of standardization in some communities, particularly in scientific domains. Probably most important, DXs directly support structures of interest in the domains they sarget, such as multi-dimensional arryas in scientific DXs. Such ordered data types are largely ignored by commercial DBMSs.

This panel will present data management issues for data managed in DX formats, present some initial approaches to these problems and try to enumerate areas where DBMS technology could do better to help.

2 Planned Participants

Dr. Serge Abiteboul, INRIA, France

Mr. Michael Achenbach, Xidak, Inc.

Dr. Robert Grossman, University of Illinois at Chicago and Design Research Institute, Cornell University

Dr. Ronald B. Melton, Battelle Pacific Northwest Laboratories

Ms. K.C. Morris, National Institute of Standards and Technology

Dr. G. Christian Overton, Human Genome Center for Chromosome 22, University of Pennsylvania

Dr. Stanely B. Zdonik, Brown University