Care and Feeding of Your PetDB

David Maier
Department of Computer Science and Engineering
OGI School of Science and Engineering at OHSU
maier@cse.ogi.edu

In 2015, for a few hundred dollars a year, you can have a personal petabyte database (PetDB) that you can access from any point of connection, with any device from a high-powered workstation to a cell phone. It stores and organizes any kind of digital data you want to have, without losing structure or information. All this data is queryable and it is arranged by type, content, structure, association and multiple categorizations and groupings. You can also locate items by when or how you encountered them, what you have done with them, where you were when you accessed them.

What could you fit in a personal petabyte store? Quite a lot, considering that if your office were filled floor to ceiling with books, that content would fit easily into 100 gigabytes. What could you have with ten thousand times more space? Some possibilities:

- The contents of every book and magazine you ever bought.
- Every web page you've ever visited.
- Every email you've ever sent or received, including all attachments.
- Every version of any piece of software you have ever used.
- All the photos you've ever taken, or received from others.
- All the notes and materials form every class, seminar, briefing and conference you've ever attended.
- All the drafts of all the documents you've ever worked on.
- Maps and images of every place you have ever been.

But there's no need to restrict yourself just to information you've "handled" in the past. You can include information you might be interested in later. For example:

- Descriptions and price lists for every kind of product you might want to buy—a "universal, personal catalog."
- Portions of the web you might want to browse, including snapshots at past points of time.
- Hotel and restaurant information on cities you plan to visit.
- Locations of gas stations and stores near the current position of your car, along with traffic conditions on roads you might be using.
- Complete historical price and performance records for any stock you might invest in.
- Portions of the web you might want to browse, including snapshots at past points of time.
- The past content of mailing lists you've joined recently.

I've called this information store a database, but that label is not totally apt. While it provides persistent and reliable storage for information, and supports expressive and efficient searching, you never have to define a "schema" for it. Your PetDB doesn't appear to reside on any particular computer—you are never on the "wrong" machine to access it. You are able to establish some level of access to it with just about any computing or communications device. Most importantly, you don't have to take any explicit action to insert data into you PetDB; your PetDB doesn't appear to have an "outside" where data is concerned. Any digital information you encounter, or express an interest in, automatically winds up in your PetDB. Your PetDB lets you query against the entirety of this information – monitoring, filtering, extracting, combining and restructuring data as needed.

The PetDB isn't predicated upon some massive improvement in holographic memory technology or DNA-based storage units. Rather, it will be built upon a new generation of software infrastructure termed Net Data Managers (NDMs). NDMs are a departure from the capabilities and structure of current database management systems. They focus on data movement, rather than data storage, working equally well with live streams of data as with files in secondary storage. They will be capable of storing data of arbitrary types, without a matching database schema having been defined previously. They will efficiently execute queries over thousands or tens of thousands of information sites. They will locate and select data items by both internal content as well as a variety of external contexts. NDMs will also support monitoring rapidly changing information sources in a way that scales to thousands or even millions of triggers.

NDM infrastructure must support the cost-effective and maintainable construction of net-centric data applications. For example, a PetDB system could be constructed by noting that (a) there is high redundancy between the contents of different PetDBs, (b) all information doesn't need to be physically present at the connecting device, if it can be brought there in reasonable time, and (c) if information was retrieved initially as the result of a query, then it can be reconstituted with the same query, if the underlying sources persist. Thus PetDB could be constructed from NDM technology linking up shared archives, existing information sources, index servers, data caches, and a relatively small amount of personal archive storage, and using a distributed query and monitoring capabilities to prestage, push and fetch on demand items "in" a particular PetDB.

Current Limitations

What are some of the limitations of current DBMSs that NDMs must overcome to support an application such as the PetDB?

A. **Schema first:** Current DBMSs require that a schema be defined before any data is stored. For the PetDB, we need to be able to store any kind of data we encounter, whether or not we have anticipated it in a schema.

B. **Load then query:** Conventional DBMSs give access to data by first loading the data into the database, then providing for queries over it. A PetDB will need to process streams of data and without a prior "load" phase.

C. **Data in the box:** With very limited exceptions, DBMS want to "own" the data they provide services over. A PetDB needs to locate, describe, index, query, and monitor data in external sources.

D. **Scale:** A PetDB may draw upon data from ten thousand different sites in a day while current distributed DBMSs can deal perhaps with a half-dozen different sources.

E. **Syntactic search:** DBMSs are well equipped for *content*-based syntactic searching. For the PetDB, we need to move up to semantic searching (to accommodate syntactic variation across sources). We also want to identify data by the *context* in which it occurs.

Research Issues

Clearly, the limitations listed above all engender areas for further research. We suggest additional topics below, some of which have already attracted research interest.

A. **Stream-processing components:** In addition to components with standard database functionality such as storage, query and indexing, NDM systems will need components specifically aimed at high-volume streams of data moving about the net. Examples are alerters (condition monitors on data streams), accumulators (conversion from stream to stored forms), semantic routers, and valence matchers (for example, to connect a "push" server to a "pull" client). The goal is to produce a handful of generic components that can be combined to implement net-centric applications.

B. **Sharing of resources:** People in the same context may want similar info; thus there is opportunity for sharing requests, processing and caching. Similar queries or subscriptions can be folded together, to avoid repeated processing or network transfers. Semantic caches that keep intensional descriptions of their content can potentially service more requests then those that keep only an extensional description. One challenge is that users operating in the same context are not necessarily connecting via the same route or through the same services, so recognizing common activities may be difficult.

C. **Privacy of data on foreign servers:** NDM applications such as the PetDB will rely on archives, repositories and caches provided by others. For example, when visiting a business, your PetDB may negotiate with a local server to pre-stage information that might be needed. It is essential that the integrity and privacy of information passing through foreign servers be maintained.

D. **Peer-to-peer sharing of context:** In an intermittently connected environment, it is important that context and recent history be able to pass directly from device to device, without needing to synchronize with some "home" server. For example, if your are accessing your PetDB via cell phone in a cab on the way to the airport, then switch to your laptop in flight, you want your laptop to directly acquire a record of recent activity from your phone.

E. **"Uncoordinated" queries:** Traditional methods for distributed query processing, with a coordinator spawning sub-queries to other sites, then combining responses into the final result, might not be suitable for all NDM applications. For example, the origin of a query and the destination for its result might be widely separated. Further, traditional distributed query processing requires that participating sites all be active and communicating at the same time. It will be useful to have alternative distributed query evaluation schemes that don't rely on a single coordinator. The goal is "distributed execution with local state."

F. **Query revision:** It must be possible to construct reasonable query answers quickly (based on nearby but possibly incomplete or stale data), then revise those answers as new information arrives or becomes accessible. One interesting problem is to characterize, based on user content and task, what constitutes a significant change in an answer (that is, one worth notifying the user about).

G. **Context capture:** Remembering the context and user activity in progress when an information item was accessed can be useful for several purposes. It can provide cues for retrieving the information later ("Find me the article I was reading on the flight to Atlanta last fall"). It can also provide hints on which data to pre-load or refresh based on a user's current activity. Such context should include both internal (application in use, other information being used, user actions) and external (time, location, other interacting users) aspects.

More Information

The Niagara Project (http://www.cs.wisc.edu/niagara/) is looking at some of the issues above in Net Data Management. Niagara is supported by DARPA through NAVY/SPAWAR Contract No. N66001-99-1-8908 and NSF ITR award IIS0086002.