l e x a n a \ n e t

February 2007

Goodbye without Leaving: The Legacy of Jim Gray

By Drew Hamre

email

As this is being written[i], legendary computer scientist Jim Gray remains missing in the Pacific Ocean. He vanished off the coast of San Francisco the evening of January 28 and - despite a massive search by the U.S. Coast Guard, the Mexican Navy, and Gray's many friends throughout the industry – not a trace has been found. After covering 132,000 square miles of ocean, the Coast Guard suspended operations the evening of February 1. Hopes for Gray's survival have dwindled since.

Gray is 63. In 1969 he received UC Berkeley's first PH.D. in Computer Science, and then began his career with UNESCO, IBM, Tandem, DEC, and Microsoft (where he's helped lead the Bay Area Research Center since 1995). Many of us know Jim Gray as a Turing Award winner and database pioneer, including early conceptual work on transactions, and software development that's at the heart of IBM's System R, Tandem's Non-Stop, and Microsoft's SQL Server. But as impressive as this record is, it's only part of the story.

Dr. Jim Gray with daughter, Heather

As others have observed, most technology firms have 'tame gurus' on staff – figureheads who are hauled out for tradeshows or for client bedazzlement. Gray, though, remains a highly regarded engineer still working at the cutting edges of technology. Gray is product-oriented, and his research foreshadows the market 5-10 years out.

His engineer's desire to build things is tempered by an academic disposition. "I aspire to be a scholar of computer science," he wrote when resigning from IBM. "All fields of scholarship (emphasize) meditation, teaching and service." Gray honors all three in his publications and talks, among the web's most generous and optimistic technology resources.

In this paper we'll review highlights from Gray's archive, touching on exabyte storage, grids, I/O optimization, and the "database revolution" he helped ignite. We'll lead with Gray's recurring concern -- the data avalanche: "How much information is there in the world?"

“How much information is there in the world?”

Michael Lesk’s informal 1997 paper - “How much information is there in the world?” - struck a chord with Jim Gray. He cites it more than fifty times, and it serves as a unifying theme for his work.

Lesk’s back-of-the-envelope calculation (now formalized by a Berkeley project) tallies the size of all digitized (or digitizable) information: web pages, the Library of Congress, home/broadcast audio/video, emails, telephony, and so on. This data avalanche is formidable (more than 10 *exabytes*[1] with 50% annual growth) and is being stored permanently: nothing is being thrown away. Moreover, most of this information will never receive human attention.

This explosion is enabled by a revolution in fast networks and cheap storage, and it impacts the entire market, not just the data center. The cost of external terabyte hard drives for home use is dropping below $500. Gray believes that within 20 years the “Personal Petabyte” will be affordable (with video comprising most of the personal data). At this same time, enterprises will have *exabytes* of information, chiefly comprised of sensor data.

However, the total ownership cost of storage remains high (with Gray citing an old data center rule-of-thumb of $300K/year per terabyte). Cooling, administration, and curation all scale with the amount of data. Because data centers and people can “afford” the storage, growth will be inevitable, and technologies must emerge to accommodate this growth.

Gray himself is deeply involved with technology’s largest data producers/consumers including the MyLifeBits project (Gordon Bell’s personal lifetime digital store), TerraServer and the Sloan Digital Sky Survey (or SDSS - digitized terrestrial and astronomical imagery), and CERN (particle physics and simulations).

Because Gray remains “hands-on” working with this information, the everyday headaches of copying, searching, and backing-up massive amounts of data are tangible problems for him, as is the necessity of their resolution. Old techniques can no longer be used, and the long-solved problems of search and transfer now require new tools (e.g., it takes roughly 2 days to GREP 1TB.)

There have been isolated successes. For example, using multi-stream TCP/IP across 10-Gbps Ethernet and OC192 links, Gray and colleagues moved almost 1GB per second between CERN (Europe) and Pasadena, CA -- a ‘land-speed record’ for such transfers. However, for domestic use the persistent problems of copying and distributing tera-scale data led Gray to revisit a “TeraScale Sneakernet”. They’ve found it’s currently simpler to ship fully populated, multi-terabyte systems than it is to transmit or otherwise copy the information.

“Free storage – It’s like free puppies!”

Data arriving at 1GB/second (as with the CERN-Pasadena test) will total ~ 30-PB/year. As Gray asks, how will we find anything? It seems unlikely that file names alone will be workable solution for managing the equivalent of 30-million files of 1GB

[1] An exabyte is 10^{18} bytes (10,000,000,000,000,000,000). It’s preceded in sequence by petabyte (10^{15}), terabyte (10^{12}, and gigabyte (10^{9}); it’s followed by zettabyte (10^{21}) and yottabyte (10^{24}).

each. The sheer amount of data means that "solved" problems must be re-visited, including (from Jim Gray's 2004 talk):

- Data capture: How do we efficiently get the bits loaded?
- Organize: How can we index the information so it's usable?
- Manage: How do we administer backups/restores/space allocations?
- Curate/Annotate: What metadata is it possible to automate?
- Keep Private: How do we assure data security?
- Summarize: Is it possible to automatically give thumbnail summaries?
- Interface: Which interfaces are best for this information (how to ask/anticipate questions)?
- Present: How is this quantity of information shown in understandable ways?

These issues may require new file system functionality, such as that currently associated with databases, including queries, indexing, pivoting, and replication.

Databases or file systems?

Jim Gray periodically revisits the question of whether databases or file systems are best suited for large data stores. Recently (December 2005), he urged a blurring of architectures … a détente between the two approaches.

Gray makes a simple differentiation. Database systems are SCHEMA FIRST – that is, metadata are defined first (e.g., tables, columns, relationships) before data are loaded. On the other hand, file systems are SCHEMA NEVER, as they offer support for little beyond a byte stream (though he notes early exceptions such as VSAM and RMS). Currently, DBMSs are the default choice for structured (columnar) data, and file systems are preferred for unstructured. "Huge battles" are raging along the borders (e.g., BLOB storage, XML just-in-time schema, and object stores)[2].

Databases and file systems own different segments of the market. As Gray notes, databases own the mid-range "sweet spot" of the market (1GB to 100TB), but the "big" data is not in databases. The biggest data stores are file based (e.g., Google and their >2-petabyte Google File System, or GFS). Overall, most bytes are in files – not databases. The web, email, and physical sciences are all file-centric.

However, with massive data stores even simple file systems are having problems organizing their directory trees. (After all, "Directories are queries," as Gray reminds his audience.) While file systems default to organizing by file name, often this is not the main dimension; rather, it's often some other piece of metadata (e.g., date or from/to/subject). Search becomes a crucial technology, along with automatic data summarization and trend/anomaly detection.

[2] In context of object stores, it would be interesting to know more about Microsoft's internal battles (technical and political) surrounding Exchange Server's *Jet Blue* implementation of its message store and SQL Server's repeated failure to supplant it.   It would also be helpful to know more about the fatal challenges faced by WinFS – promoted in the early 1990s (Cairo), then deferred to become the promised centerpiece of Longhorn, then relegated to an add-on, and now all but dead with its obituary appearing in an MSDN blog entry.

The file system/DBMS détente envisioned by Gray would let a file look like a database table/XML doc, while letting a database table/XML doc look like a file. Systems would support both interfaces, optionally supporting transactional semantics. Schemas would be extracted from objects on a just-in-time basis. The data store would be covered by lots of indices. Gray cites Win/FS as an example of how this blurring of technologies might work (though this may be an unfortunate choice of examples, as the future of WinFS is in doubt; see *Footnote 2*).

Although not explicitly cited, Google itself may be an example of this détente, as their file content (stored on GFS) is transparently supplemented by a custom DBMS ("Bigtable") which handles semi-structured data, including URLs and per-user data.

Storing the bits of Gordon Bell's life

Jim Gray's equally famous colleague, Gordon Bell, has been building a lifetime digital repository over the past several years. In its current implementation, MyLifeBits is implemented as a SQL Server database that stores content and metadata for items such as contacts, documents, email, events, photos, songs, and video. Items have about 20 common properties, and are extended by attributes specific to the item type (e.g., contacts have an additional 62 attributes).

As the system matures (it's been operational for five years), its original folder-hierarchy organization has been abandoned. In effect, "everything can be in one, large folder and items are retrieved by their content with no attention to 'location' ... Items can be linked together implicitly [by time or geographic proximity] … or explicitly." The current design offers freedom from a strict storage hierarchy, as well as offering full text search, some automatic annotations, and many visualizations (e.g., a mapping UI for GPS records, a day-planner view of activity logs).

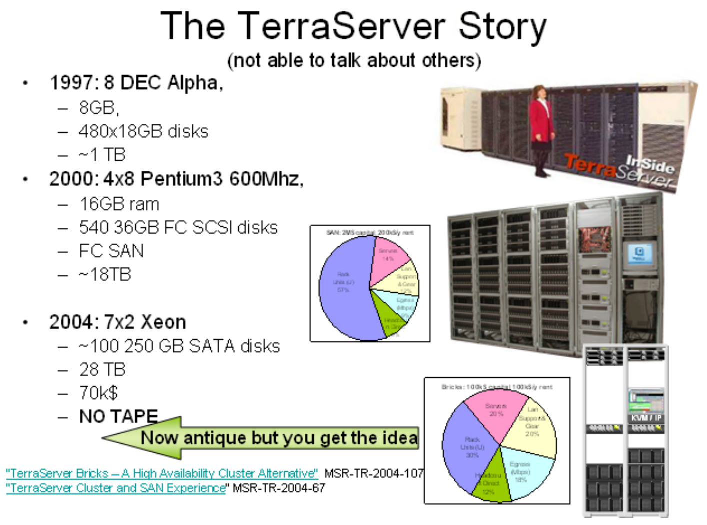
MyLifeBits items are grouped in generic collections, and items can be queried, pivoted, and correlated using various metadata attributes. The power of this unified experiential store is valued by Bell for everyday tasks, such as "finding a particular tile model that was used in a 5 year old home renovation by retrieving the contractor's specifications and invoice; (or) … using a caller name to identify a particular call time to retrieve a web page being viewed at that time."

The current size of Gordon Bell's SQL Server-based store is ~110-GB; however, projections for a full (85-year) span exceed 200-TB … and thus begin approaching the 'personal petabyte' both predicted and feared by Jim Gray.

Direct attached disks ('storage bricks') or SANs?

Gray's *TeraScale SneakerNet* is an example where direct attached disks, or 'DADs', are beneficial. Gray contrasts DADs with currently-favored SAN architectures, noting that SANs tend to have high initial costs (Gray estimates ~$3K/spindle or $20K per TB), high administrative costs, and lengthy configuration/tuning cycles. Gray also found SANs could have slower I/O performance compared to other storage solutions, at least where I/O tended to favor sequential workloads (e.g., data warehousing and mining).

Gray cites a recent TPC-H benchmark using DADs rather than a SAN. The DL585 (4 dual core AMD Opterons with 64GB) was configured with 3-TB of SAS storage (80 disks, connected to 8 P600s). At the time results were published, the system was both the performance and price/performance leader. Cost for the DAD-based solution was roughly \$110K, versus \$820K for the runner-up.



The TerraServer Story

(not able to talk about others)

- 1997: 8 DEC Alpha,
  - 8GB,
  - 480x18GB disks
  - ~1 TB
- 2000: 4x8 Pentium3 600Mhz,
  - 16GB ram
  - 540 36GB FC SCSI disks
  - FC SAN
  - ~18TB
- 2004: 7x2 Xeon
  - ~100 250 GB SATA disks
  - 28 TB
  - 70k$
  - NO TAPE

Now antique but you get the idea

"TerraServer Bricks – A High Availability Cluster Alternative" MSR-TR-2004-107
"TerraServer Cluster and SAN Experience" MSR-TR-2004-67

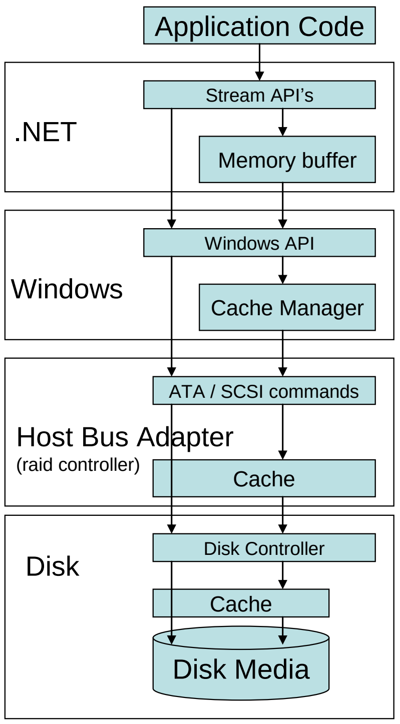
Gray notes that major Internet presences such as Google and Yahoo! use DADs. Microsoft's MSN and Hotmail were originally SAN-based, but by 2004 were either making or had completed the transition to DADs. As for TerraServer (the SQL Server-based satellite photo archive) it was originally SAN-based, but in 2004 it also transitioned to storage bricks.

However, Gray still acknowledges SANs as superior in their ability to detect and handle failures transparently, to scale, and for fast random I/O.

The world speed record for Windows sequential I/O

In 2004, Jim Gray and Peter Kukol revisited earlier benchmarks of the fastest possible sequential I/O under Windows. The progress has been impressive:

- In 1997, sophisticated, low-level programming tricks were necessary to read/write disks at 7MBps on Windows.
- In 2000, non-standard programming was still needed to drive the faster hardware to its potential, and reached 24 MBps of read throughput.
- In the latest (2004) study, the key finding for developers is that out-of-the-box .NET performance is excellent. In fact, "a file striped across 16 disk drives delivers 800 MBps and uses about 30% of a processor."



Between each iteration of the test, processors, disks, and controllers all improved enormously. However, Gray and Kukol lay much of the improvement to the software stack, which "has been streamlined and redesigned so that it usually does the right thing". (The sidebar figure is from their paper, and presents the hardware/software and caching layers in the .NET disk I/O path.)

Based on these findings, Gray & Kukol recommend:

- For single disks, use the defaults of the .NET Framework, as they deliver excellent sequential performance.
- Pre-allocate large sequential files (`SetLength()`). This improves speed by about 13% when compared to a fragmented file.
- Surprisingly, disk arrays required *un*-buffered I/O to achieve the highest performance (as buffered I/O can be 8x slower). Gray & Kukol expect this to be addressed in subsequent versions of.NET.

In addition to the test's benchmark results and analysis, all C# source code is also available. The code library includes several programs, including: *IOexamples* (samples of different techniques for sequential I/O in .NET); *AsyncIO* (sample of unbuffered AsyncIO); and *IOspeed* (which measures the speed of disk file access).

Gray & Kukol summarize technology's progress as follows. In 1997, you needed deep I/O development skills to drive disks to their potential, and the bottlenecks were controllers and busses. In 2000, controllers and busses remained the bottlenecks. By 2004, the .NET software stack delivered excellent performance by default, and great results were obtained with commodity SATA disks and controllers.

WAN clusters: When do computing grids make sense?

Jim Gray characterizes grid-scale distributed computing as *WAN rather than LAN-clusters*. Always comfortable using macro-economic trends to drive technology forecasts, Gray in 2003 published an analysis of grid computing driven by economics.

Though cost ratios are volatile, in 2003 there was rough parity in the cost of "(1) one database access, (2) ten bytes of network traffic, (3) 100,000 instructions, (4) 10 bytes of disk storage, and (5) a megabyte of disk bandwidth". Beginning with some striking anecdotes about free and ad-supported computing grids (SETI@Home delivers 54 teraflops; Google handles a trillion searches/year against the world's largest online data store (2 petabytes); Hotmail carries a trillion email messages/ year), Gray works through the implications of how to best to realize the benefits of such distributed computing networks.

Current economics argue that it's necessary to "put computing as close to the data as possible in order to avoid expensive network traffic" (though as Gray later stresses, "If telecom prices drop faster than Moore's law, the analysis fails. If telecom prices drop slower than Moore's law, the analysis becomes stronger."). Gray offers the following rule of thumb:

- "It seems the computation should be at least 30,000 instructions per byte (a 3:1 cost benefit ratio) before the outsourcing model becomes really attractive. The break-even point is 10,000 instructions per byte of network traffic or about a minute of computation per MB of network traffic."

Which applications are best suited for a distributed infrastructure?

The likeliest grid-candidates are those applications that are "mobile and that can be provisioned on demand" – specifically purely computational tasks "written in a portable language and (using) only portable interfaces". The ideal mobile task is "stateless (no database or database access), has a tiny network input and output, and has huge computational demand".

SETI@Home is seems ideally suited for this architecture, distributing roughly a billion jobs of .5-MB each, which then each computed for roughly 12 hours on this half-megabyte of input. As Gray summarizes, the key about SETI@Home is its "ComputeCost:NetworkCost ratio is 10,000:1. It is very cpu-intensive."

Gray sees many other classes of problems suited to distributed grids: cryptography, risk simulation, and graphic rendering are among those cited. For example, "Pixar's *Toy Story 2* images are very CPU intensive – a 200 MB image can take several CPU hours to render. The instruction density was 200k to 600k instructions per byte. This could be structured as a grid computation – sending a 50MB task to a server that computes for ten hours and returns a 200MB image."

The class of applications with which Gray is most often associated -- data loading and data scanning -- are CPU-intensive, but also data intensive. At this time, they're not economically viable for distributed architectures. However, Gray and a group of astronomers used SDSS data to show the benefits of distributed database processing. As they summarized, "It is a mistake to move large amounts of data to the query, when you can move the query to the data and execute the query in parallel. For this reason, it would be useful for nodes on the Grid to support different Database Management Systems so that SQL applications could be deployed as easily as traditional Grid applications coded in C, Fortran, etc."

The revolution in database architecture

In early 2004, Gray summarized the changes in database functionality he's seen during his career. Gray listed 13 major advances, including object-relational databases, web service integration, queues/transactions/ workflows, cubes and OLAP, semi-structured data, stream processing, data mining, and so on.

Taken in aggregate, Gray argued that these changes were revolutionary. Databases were becoming self-sustaining (always-on), were encapsulating both code and data, and were moving to the periphery of the network. "Database systems are becoming database operating systems, into which one can plug subsystems and applications. … Each disk and each sensor will be a competent database machine. Relational algebra is a convenient way to program these systems. … The restructuring of database systems to be web services and to integrate with language runtimes has created a modularity that enables these revolutions. The reunification of code and data is pivotal. Almost all the other changes depend on that."

In a related talk, Gray said we should no longer be asking which objects should be added to databases; instead we should ask "What kind of object is a database?" In today's products, stored procedures have evolved to "real" languages; data can be encapsulated, and exposed as a class with methods; records are vectors of objects; and transactions can operate even on these objects.

As Gray observes, smart devices mean databases are everywhere: phones, PDAs, and cameras all have small databases inside, and these devices all need to share data. However, much work remains before these distributed data stores are so robust that they never lose data, they interoperate, and they always answer questions efficiently.

The mystery of Jim Gray's disappearance

The Sunday morning of January 28 was clear and calm when Jim Gray left San Francisco in his yacht, *Tenacious*. He intended to sail across 27-miles of ocean to the

Farallon Islands following a well-traveled commercial channel. Half the distance is marked by buoys, and Gray had sailed this route many times before.

Gray called his daughter, Heather, about 10:30AM to say he was sailing out of cell-phone range, but that it was a beautiful day and that he'd spotted dolphins. There was no further contact. Gray's wife reported him missing at 8:35 that evening.

A Coast Guard officer called Gray's disappearance "a mystery" because of the good weather, Gray's good health, and the excellent condition of the boat and its equipment. There was no distress signal, no sign of wreckage, and later inspection revealed no collision damage to freighters navigating the channel.

The government's search was aided by Gray's friends throughout the industry – not only at Microsoft, but also at Amazon (Gray was on CTO Werner Vogels' dissertation committee), NASA (who lent a spy plane), and Google (where co-founder Sergey Brin sought to apply *Google Earth* imagery to the search).

Ten years ago, super-computer pioneer Seymour Cray was kept alive an additional 2 weeks following a traffic accident thanks to medical technology designed on his systems. With Jim Gray, the irony is repeated as massive stores of graphical data are being distributed and collaboratively analyzed in the search.

"This one is unusual," the Coast Guard officer said. "We have no clues.'' RADARSAT images are about 12.5 meters per pixel, and the 40-foot Tenacious would be a single dot amidst the blue. And wherever he is, Jim Gray would know better than most: it is an indescribably vast ocean.

Additional Resources

Summary Home Page – Jim Gray's Research Portal
http://research.microsoft.com/%7EGray/

Turing Award Lecture
http://research.microsoft.com/%7EGray/talks/Turing2.ppt

"Behind the Code - Conversation with scientist and legend Jim Gray" -- MSDN Video
http://channel9.msdn.com/ShowPost.aspx?PostID=168181

Tenacious Search Blog (Note: suspended on February 16, 2007)
http://www.openphi.net/tenacious/

Acknowledgements

Credits – The introductory photograph of Jim Gray and his daughter Heather is from a search handout, reprinted in the San Jose Mercury News. *Goodbye Without Leaving* is the title of a novel by Laurie Colwin.

Thanks to reviewers for their time and suggestions; any inaccuracies remain mine.

Summary

Turing Award-winning computer scientist Jim Gray has been missing in the Pacific since January 28. Famed as the father of Microsoft's SQL Server, Gray also contributed seminal work on transactions, as well as software development at the core of IBM's System R and Tandem's Non-Stop. Gray's research is both pragmatic and far-ranging, touching on computing grids, exabyte data management, I/O

optimization, and the "database revolution" he helped ignite. Gray's recurring concern is the data avalanche: "How much information is there in the world?"

This paper has shared highlights from Gray's archive in hopes of encouraging others to read these works. Any well-meaning summary (including this paper) does Gray a disservice, because only his original writings capture the breadth of curiosity, the rigor of analysis, the utility of the results, and the joy he found studying computer science. Gray's joy is contagious, and it will brighten your day to visit his library.

Drew Hamre is a principal of lexana\net and lives in Golden Valley, Minnesota.

---

[i] On May 31, 2008, Berkeley University hosted a *Tribute to Honor Jim Gray* ('tribute' rather than 'memorial' or 'funeral', because the fates of Jim Gray and his boat *Tenacious* remain unknown).

More than 1,000 people attended the event, which received widespread media coverage that was both laudatory and affectionate. "There are a lot of smart people at Microsoft. (Jim Gray) was the one person who everybody thought was smarter than they were," said David Vaskevitch, senior vice president at Microsoft (quoted in InformationWeek).

"He was one of the world's great listeners," said Ed Lazowska, a University of Washington computer scientist. "I thought we had a special relationship," he said, only to realize that there were 500 special relationships of the same kind (quoted on the New York Times web site).

Shankar Sastry, dean of Berkeley's College of Engineering, noted that Gray helped found the Berkeley Water Center and regularly worked with faculty and students. Gray "was a wonderful guy … a towering intellect, but (he) still had time to advise (others)".

The re-tellings of Gray's accomplishments were leavened with humor. David DeWitt (who recently started Microsoft's Jim Gray Systems Lab in Madison, WI), remembered how Gray strove to get his seminal paper on transaction benchmarks published on April 1, and that the last line was "There are lies, damn lies and then there are performance measures."

Gray's friend and fellow computing legend, Gordon Bell, gently poked fun at some of the no-doubt over-blown praise: "Then of course there's Gray matter; we've named a part of the brain after Jim." In his "funny and touching talk", Bell spoke about Gray as a connectionist who knew no boundaries and of his paradigm for data exploration. As Seattle columnist Brier Dudley summarized, Gray's approach "mixes computer science with other sciences, unifying theory, experiment and simulation. It involves huge amounts of data captured by instruments and simulation, software to process it and information or knowledge that's stored in computers." Thus, said Gordon Bell, "computer scientists really learn the scientists' science and become a co-partner" in the work.

In one of the most intensive searches ever off San Francisco, both the Coast Guard and a family-sponsored search recovered no debris or other evidence of an accident. Side scanning sonar was used, mapping undersea rocks and previously-known (and unknown) wrecks, but providing no trace of the *Tenacious*. The family has donated search data to governmental and nonprofit organizations. Technical proceedings from the 'Tribute' will be published in a special issue of an ACM publication.

UC Berkeley administrators announced that Google, Microsoft, Oracle and others are endowing a Jim Gray chair in Computer Science.