How Best to Build Web-Scale Data Managers? A Panel Discussion

Philip A. Bernstein (Microsoft), Daniel J. Abadi (Yale),
Michael J. Cafarella (U. of Washington), Joseph M. Hellerstein (U.C. Berkeley),
Donald Kossmann (ETH Zürich), Samuel Madden (M.I.T.)

1. PANEL OVERVIEW

Many of the largest database-driven web sites use custom web-scale data managers (WDMs). On the surface, these WDMs are being applied to problems that are well-suited for relational database systems. Some examples are the following:

- Map-Reduce [5], Hadoop [7], and Dryad [9] are used to process queries on large data sets using sequential scan and aggregation. Hive [8] is a data warehouse built on Hadoop.
- Google's Bigtable [3] is used to store a replicated table of rows of semi-structured data.
- Amazon's Dynamo [6] is used to store partitioned, replicated databases of key-value pairs. Cassandra [2] is similar.
- Object caching systems are used instead of a persistent store, such as memcached [10], Oracle's Coherence, and Microsoft's Velocity project.

These WDMs have challenging requirements that are not met by current relational database products. They need to scale out to thousands of machines, offer high availability even on unreliable commodity hardware, and be completely self-managing. To make it easier to meet these requirements, these WDMs offer much less functionality than a relational database system. Yet the functionality is apparently enough to attract a wide following.

The differences between these WDMs and relational database systems are striking. This panel will explore these differences. In particular, it address the following questions:

- What should the database field be doing to satisfy the needs of web-scale data management?
- Many web-scale WDMs were built primarily by systems groups whose specialty is not classical database management. (One exception is PNUTS [4].) What does this say about the database field? What should we be doing differently?
- Do web-scale data management problems require very limited functionality to satisfy other requirements? Or is this just a symptom of immature technology that will improve?
- Many of these WDMs abandon ACID transactions and require the application to deal with data consistency. Is this the only hope to achieve satisfactory scale-out?
- Many developers prefer these limited-functionality WSDMs to classical DBMSs. Why? How do we increase functionality without sacrificing ease of use?
- Is it practical to obtain a competitive WDM by improving the scalability, availability and manageability of a classical DBMS (as in [1])?

VLDB '09, August 24-28, 2009, Lyon, France.

2. ACKNOWLEDGMENTS

We are grateful to Sergey Melnik for proposing the panel and to Chris Olston for advice on issues to address.

3. REFERENCES

[1] P. A. Bernstein, N. Dani, B. Khessib, R. Manne, D. Shutt: Data Management Issues in Supporting Large-Scale Web Services. IEEE Data Eng. Bull. 29(4): 3-9 (2006)

[2] http://incubator.apache.org/cassandra/

[3] F. Chang, J. Dean, S. Ghemawat, W.C. Hsieh, D.A. Wallach, M. Burrows, T. Chandra, A. Fikes, R.E. Gruber: Bigtable: A Distributed Storage System for Structured Data. ACM Trans. Comput. Syst. 26(2): (2008)

[4] B. F. Cooper, R. Ramakrishnan, U. Srivastava, A. Silberstein, P. Bohannon, H-A Jacobsen, N. Puz, D. Weaver, R. Yerneni: PNUTS: Yahoo!'s hosted data serving platform. PVLDB 1(2): 1277-1288 (2008)

[5] J. Dean, S. Ghemawat: MapReduce: Simplified Data Processing on Large Clusters. OSDI 2004: 137-150

[6] G. DeCandia, D. Hastorun, M.Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, W. Vogels: Dynamo: Amazon's highly available key-value store. SOSP 2007: 205-220

[7] Hadoop. http://lucene.apache.org/hadoop/.

[8] Hive. http://wiki.apache.org/hadoop/Hive.

[9] M. Isard, M. Budiu, Y. Yu, A. Birrell, D. Fetterly: Dryad: distributed data-parallel programs from sequential building blocks. EuroSys 2007: 59-72

[10] http://www.danga.com/memcached