TAGS

Basho, Big Data, Big Table, Dynamo, Erlang, eventual consistency, Hadoop, Justin Sheehy, MapReduce, MongoDB, NewSQL, NoSQL, nosql databases, open source, Riak, Werner Vogels

On Eventual Consistency– An interview with Justin Sheehy.

by Roberto V. Zicari on August 15, 2012

***"I would most certainly include updates to my bank account as applications for which eventual consistency is a good design choice. In fact, bankers have understood and used eventual consistency for far longer than there have been computers in the modern sense"* –Justin Sheehy.**

On the subject of new data models and eventual consistency I did interview **Justin Sheehy** *Chief Technology Officer, **Basho Technologies.***

RVZ

Q1. What are in your opinion the main differences and similarities of a key-value store (ala Dynamo), a document store (ala MongoDB), and an "extensible record" store (ala Big Table) when using them in practice?

Justin Sheehy: Describing the kv-store, doc store, and column family data models in general is not the same as describing specific systems like Dynamo, MongoDB, and BigTable. I'll do the former here as I am guessing that is the intention of the question. Since the following couple of questions ask for differences, I'll emphasize the similarity here.

All three of these data models have two major things in common: values stored in them are not rigidly structured, and are organized mainly by primary key. The details beyond those similarities, and how given systems expose those details, certainly vary. But the flexibility of semi-structured data and the efficiency of primary-key access generally apply to most such systems.

Q2. When is a key-value store particularly well suited and when is a a document store instead preferable? For which kind of applications and for what kind of data management requirements?

Justin Sheehy: The interesting issue with this question is that "document store" is not well-established as having a specific meaning. Certainly it seems to apply to both MongoDB and CouchDB, but those two systems have very different data access semantics. The closest definition I can come up with quickly that covers the prominent systems known as doc stores might be something like "a key-value store which also has queryability of some kind based on the content of the values."

If we accept that definition then you can happily use a document store anywhere that a key-value store would work, but would find it most worthwhile when your querying needs are richer than simply primary key direct access.

Q3. What is Riak? A key-value store or a document store? What are the main features of the current version of Riak?

Justin Sheehy: Riak began as being called a key-value store before the current popularity of the term "document store" term, but it is certainly a document store by any reasonable definition that I know — such as the one I gave above. In addition to access by primary key, values in Riak can be queried by secondary key, range

About the Author

Roberto V. Zicari

Prof. Roberto V. Zicari is editor of **ODBMS.ORG (www.odbms.org)** , the "most up-to-date collection of free materials on object database technology on the Internet". The portal was created to serve software developers in the open source community or at commercial companies as well as faculty and students at educational and research institutions.

ODBMS.ORG is designed to meet the fast-growing need for resources focusing on Big Data, Analytical Data Platforms, Scalable Cloud platforms, Object databases, Object-relational bindings, NewSQL databases, NoSQL datastores, and new approaches to concurrency control.

Roberto is Full Professor of Database and Information Systems at Frankfurt University and representative of the OMG in Europe. Previously, Roberto served as associate professor at Politecnico di Milano, Italy; Visiting scientist at IBM Almaden Research Center, USA, the University of California at Berkeley, USA; Visiting professor at EPFL in Lausanne, Switzerland, the National University of Mexico City, Mexico and the Copenhagen Business School, Danemark.

Search

Tags

Analytics award awards Big Data Caché cloud stores common persistent model patterns database design patterns db4o db4objects document stores Goethe University Frankfurt Google Google BigTable Hadoop IBM ICOODB impedence mismatch International Conference on Object Databases International Conference on Objects and Databases InterSystems Java Java Object Persistence Lecture Notes MapReduce Michael Blaha MongoDB

query, link walking, full text search, or map/reduce.

Riak has many features, but the core reasons that people come to Riak over other systems are Availability, Scalability, and Predictability. For people whose business demands extremely high availability, easy and linear scalability, or predictable performance over time, Riak is worth a look.

Q4. How do you achieve horizontal scalability? Do you use a "shared nothing" horizontal scaling – replicating and partitioning data over many servers?
What performance metrics do you have for that?

Justin Sheehy: We use a number of techniques to achieve horizontal scalability. Among them is consistent hashing, an approach invented at **Akamai** and successfully used by many distributed systems since then. This allows for constant time routing to replicas of data based on the hash of the data's primary key.
Data is partitioned to servers in the cluster based on consistent hashing, and replicated to a configurable number of of those servers. By partitioning the data to many "virtual nodes" per host, growth is relatively easy as new hosts simply (and automatically) take over some of the virtual nodes that has previously owned by existing cluster hosts.
Yes, in terms of data location Riak is a "shared nothing" system.
One (of many) demonstrations of this scalability was performed by Joyent **here**.
That benchmark is approximately 2 years old, so various specific numbers are quite outdated, but the important lesson in it remains and is summed up by this graph late in this **post.**
It shows that as servers were added, the throughput (as well as the capacity) of the overall system increased linearly.

Q5. How do you handle updates if you do not support ACID transactions? For which applications this is sufficient, and when this is not?

Justin Sheehy: Riak takes more of the "**BASE**" approach, which has become accepted over the past several years as a sensible tradeoff for high-availability data systems. By allowing consistency guarantees to be a bit flexible during failure conditions, a Riak cluster is able to provide much more extreme availability guarantees than a strictly ACID system.

Q6. You said that Riak takes more of the "BASE" approach. Did you use the definition of eventual consistency by Werner Vogels?
Reproduced here: "Eventual consistency: The storage system guarantees that if no new updates are made to the object, eventually (after the inconsistency window closes) all accesses will return the last updated value". You would not wish to have an "eventual consistency" update to your bank account. For which class of applications is eventual consistency a good system design choice?

Justin Sheehy: That definition of Eventual Consistency certainly does apply to Riak, yes.

I would most certainly include updates to my bank account as applications for which eventual consistency is a good design choice. In fact, bankers have understood and used eventual consistency for far longer than there have been computers in the modern sense. Traditional accounting is done in an eventually-consistent way and if you send me a payment from your bank to mine then that transaction will be resolved in an eventually consistent way. That is, your bank account and mine will not have a jointly-atomic change in value, but instead yours will have a debit and mine will have a credit, each of which will be applied to our respective accounts.

This question contains a very commonly held misconception. The use of eventual consistency in well-designed systems does not lead to inconsistency. Instead, such systems may allow brief (but shortly resolved) discrepancies at precisely the moments when the other alternative would be to simply fail.

MySQL New and old Data stores NoSQL nosql databases object databases object persistence ODBMS ODBMS.ORG ODBTWG OMG On Innovation open source ORM relational databases SQL standards Versant VoltDB

Archives

November 2012
October 2012
September 2012
August 2012
July 2012
June 2012
May 2012
April 2012
March 2012
February 2012
January 2012
December 2011
November 2011
October 2011
September 2011
August 2011
July 2011
June 2011
May 2011
April 2011
March 2011
February 2011
January 2011
December 2010
November 2010
October 2010
September 2010
August 2010
July 2010
June 2010
May 2010
April 2010
March 2010
February 2010
January 2010
December 2009
November 2009
October 2009
September 2009
August 2009

To rephrase your statement, you would not wish your bank to fail to accept a deposit due to an insistence on strict global consistency.

It is precisely the cases where you care about very high availability of a distributed system that eventual consistency might be a worthwhile tradeoff.

Q7. Why is Riak written in Erlang? What are the implications for the application developers of this choice?

Justin Sheehy: Erlang's original design goals included making it easy to build systems with soft real-time guarantees and very robust fault-tolerance properties. That is perfectly aligned with our central goals with Riak, and so Erlang was a natural fit for us. Over the past few years, that choice has proven many times to have been a great choice with a huge payoff for Riak's developers. Application developers using Riak are not required to care about this choice any more than they need to care what language PostgreSQL is written in. The implications for those developers are simply that the database they are using has very predictable performance and excellent resilience.

Q8. Riak is open source. How do you engage the open source community and how do you make sure that no inconsistent versions are generated?

Justin Sheehy: We engage the open source community everywhere that it exists. We do our development in the open on github, and have lively conversations with a wider community via email lists, IRC, Twitter, many in-person venues, and more. Mark Phillips and others at Basho are dedicated full-time to ensuring that we continue to engage honestly and openly with developer communities, but all of us consider it an essential part of what we do. We do not try to prevent forks. Instead, we are part of the community in such a way that people generally want to contribute their changes back to the central repository. The only barrier we have to merging such code is about maintaining a standard of quality.

Q9. How do you optimize access to non-key attributes?

Justin Sheehy: Riak stores index content in addition to values, encoded by type and in sorted order on disk. A query by index certainly is more expensive than simply accessing a single value directly by key, as the indices are distributed around the cluster — but this also means that the size of the index is not constrained by a single host.

Q10. How do you optimize access to non-key attributes if you do not support indexes in Riak?

Justin Sheehy: We do support indexes in Riak.

Q11 How does Riak compare with a new generation of scalable relational systems (NewSQL)?

Justin Sheehy: The "NewSQL" term is, much like "NoSQL", a marketing term that doesn't usefully define a technical category. The primary argument made by NewSQL proponents is that some NoSQL systems have made unnecessary tradeoffs. I personally consider these NewSQL systems to be a part of the greater movement generally dubbed NoSQL despite the seemingly contradictory names, as the core of that movement has nothing to do with SQL — it is about escaping the architectural monoculture that has gripped the commercial database market for the past few decades. In terms of technical comparison, some systems placing themselves under the NewSQL banner are excellent at scalability and performance, but I know of none whose availability and predictability can rival Riak.

Q12 Pls give some examples of use cases where Riak is currently in use. Is Riak in use for analyzing Big Data as well?

Justin Sheehy: A few examples of companies relying on Riak in their business can be found **here.**
While Riak is primarily about highly-available systems with predictable low-latency

July 2009
June 2009
May 2009
April 2009
March 2009
February 2009
January 2009
December 2008
November 2008
October 2008
September 2008
August 2008
July 2008
June 2008
May 2008
April 2008
March 2008
February 2008
January 2008
December 2007
November 2007
October 2007
September 2007
August 2007
April 2006
February 2006
September 2005

Meta

Log in
Entries RSS
Comments RSS
WordPress.org

performance, it does have analytical capabilities as well and many users make use of map/reduce and other such programming models in Riak. By most definitions of "Big Data", many of Riak's users certainly fall into that category.

Q Anything you wish to add?

Justin Sheehy: Thank you for your interest. We're not done making Riak great!

_____

Justin Sheehy
Chief Technology Officer, Basho Technologies

As Chief Technology Officer, Justin Sheehy directs Basho's technical strategy, roadmap, and new research into storage and distributed systems.
Justin came to Basho from the MITRE Corporation, where as a principal scientist he managed large research projects for the U.S. Intelligence Community including such efforts as high assurance platforms, automated defensive cyber response, and cryptographic protocol analysis.
He was central to MITRE's development of research for mission assurance against sophisticated threats, the flagship program of which successfully proposed and created methods for building resilient networks of web services.
Before working for MITRE, Justin worked at a series of technology companies including five years at Akamai Technologies, where he was a senior architect for systems infrastructure giving Justin a broad as well as deep background in distributed systems.
Justin was a key contributor to the technology that enabled fast growth of Akamai's networks and services while allowing support costs to stay low. Justin performed both undergraduate and postgraduate studies in Computer Science at Northeastern University.

_____

Related Posts

- **On Data Management: Interview with Kristof Kloeckner, GM IBM Rational Software.**

- **On Big Data: Interview with Dr. Werner Vogels, CTO and VP of Amazon.com**

Resources

ODBMS.org — Free Downloads and Links

In this section you can download resources covering the following topics:
Big Data and Analytical Data Platforms
Cloud Data Stores
Object Databases
NewSQL,
NoSQL Data Stores
Graphs and Data Stores
Object-Oriented Programming
Entity Framework (EF) Resources
ORM Technology
Object-Relational Impedance Mismatch
XML, RDF Data Stores,
RDBMS

##

From → uncategorized

2 Comments Leave one →

Mirco Bianco permalink

In my opinion, the distinguishing feature of Riak is the soft real-time as evidenced in Q7. Riak, due to its soft real-time capabilities, could be also employed in financial applications like financial intelligence applications, wherein it necessary to fulfill time-contraints imposed by the stock markets in order to success in the business. This is probably done by Trifork (https://www.trifork.com/), one of the Basho's customers. Correct?

Reinhold Thurner, permalink

On "banking and eventual consistency":
I was directly involved in the payment system of a bank and must say that "eventual consistency" would certainly not be an option. I am used to look at subjects from a conceptual view of the "things" and their relationship (no surprise – since my primary interest is Metasafe): A funds transfer affects four accounts (mine, myBanks, hisBanks, his), a payment order and needs three consistent transactions – independent whether it is made by a computer or manually on paper.

I wonder if "eventual consistency" means AeCID or AID or just D? I always thought that a system is either ACID or it is not (like being pregnant). Michael Stonebraker says (and explains) in his keynote (http://www.youtube.com/watch?v=uhDM4fcI2aI) on NewOLTP and NewSQL – that "eventual consistency may create garbage".

In this presentation Michael uses the term "newSQL" as a name for the architecture of an in core relational DBMS in contrast to the in core emulation of a disk based relational DBMS. It is not that: "The "NewSQL" term is, much like "NoSQL", a marketing term that doesn't usefully define a technical category".

"Precisely where concepts fail you, A timely word will come to mind." (GOETHE, Faust I:1995–1996). Fiddling with words can create the impression (for users) that ACID may be turned into a sort of jelly and endanger our credibility. I can accept that a superior level of availability means no guaranteed consistency. We have to accept that "One size does not fit all" (Stonebraker). A taxonomy (see page 189 in the paper of David Maier http://www.odbms.org/download/clouddbms2011.pdf) and a true and fair description about the properties and tradeoffs would help to position new solutions and to make the right choice in any given situation.

Leave a Reply

Name (required)

Email (required, will not be published)

Website

Comment

Note: HTML is allowed. Your email address will not be published.

Subscribe to this comment feed via RSS

Submit Commen

Spam protection by WP Captcha-Free

About

"This is the ODBMS Industry Watch blog. --Trends and Information on New Data Management Technologies, Innovation."

To see the complete ODBMS website with useful articles, downloads and industry information, please CLICK HERE.

August 2012

M	T	W	T	F	S	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

« Jul Sep »

Search

Search