TAGS

ACID, ACID transactions, Analytics, BASE, Basho Technologies, Big Data, CAP theorem, Justin Sheehy, MariaDB, Michael "Monty" Widenius, MySQL, NoSQL, nosql databases, open source, Riak

On Eventual Consistency– Interview with Monty Widenius.

by Roberto V. Zicari on October 23, 2012

"For analytical things, eventual consistency is ok (as long as you can know after you have run them if they were consistent or not). For real world involving money or resources it's not necessarily the case." — Michael "Monty" Widenius.

In a **recent interview, I asked Justin Sheehy**, Chief Technology Officer at Basho Technologies, maker of **Riak**, the following two questions, related to the subject of **eventual consistency**:

*Q1. "How do you handle updates if you do not support **ACID** transactions? For which applications this is sufficient, and when this is not?"*

*Q2. "You said that Riak takes more of the "**BASE**" (Basically Available, Soft state, Eventual consistency) 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? "*

On the same subject, I did a follow up interview with **Michael "Monty" Widenius**, the main author of the original version of the open-source **MySQL** database, and currently working on a branch of the MySQL code base, called **MariaDB**.

RVZ

Q. Justin Sheehy`s reply to Q1: *"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."*
When do you think a "BASE" approach to consistency is justified?

"Monty" Widenius: The big questions are:
a) How are conflict's solved? Who will win when there are conflicting updates on two nodes and the communication between the nodes are temporarily down?
b) Can a user at any point read data that is not consistent?
c) How long can the conflicting window be?

The answers to the above questions tells us how suitable the solution is for different applications. For analytical things, eventual consistency is ok (as long as you can know after you have run them if they were consistent or not). For real world involving money or resources it's not necessarily the case.

Q. How do you handle consistency in MariaDB and at the same time ensuring scalability and availability? Aren't you experiencing the limitations of the CAP Theorem?

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

"Monty" Widenius: We are using the traditional approaches with transactions or synchronous replication when you need guaranteed consistent answers.

We also provide asynchronous updates to slaves when you can tolerate a log for the data on the slaves. However, as we are only making things visible when the total transaction is run on either master/slave you have always things consistent.

So when it comes to CAP, it's up the user to define where he wants to have his tradeoff; Speed, reliability or easy to manage.

Q. Justin Sheehy`s reply to Q2: *"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."*

"Monty" Widenius: The question is time spent between the consistency and where things will be inconsistent. For example, at no point in time should there be more money on my account than I have the right to use.
The reason why banks in the past have been using eventual consistency is that the computer systems on the banks simply has not kept up with the rest of the word.
In many places there is still human interaction needed to get money on the account! (especially for larger amounts).
Still, if you ask any bank, they would prefer to have things always consistent if they could!

Q. Justin says that *"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".*

"Monty" Widenius: In some cases it's better to fail. For example it's common that ATM will not give out money when the line to the bank account is down. Giving out money is probably always the wrong choice. The other question is if things are 100 % guaranteed to be consistent down to the millisecond during normal operations.

Q. Justin says: *"to rephrase your statement, you would not wish your bank to fail to accept a deposit due to an insistence on strict global consistency."*

"Monty" Widenius: Actually you would, if you can't verify the identity of the user. Certainly the end user would not want to have the deposit be accepted if there is only a record in a single place of the deposit.

Q. Justin says: *"It is precisely the cases where you care about very high availability of a distributed system that eventual consistencymight be a worthwhile tradeoff."*
What is your take on this? Is Eventual Consistency a valid approach also for traditional banking applications?

"Monty" Widenius: That is what banks have traditionally used. However, if they would have a choice between eventual consistency and always consistent they would always choose the later if it would be possible within their resources.

———————

Michael "Monty" Widenius is the main author of the original version of the open-source MySQL database and a founding member of the MySQL AB company. Since 2009, Monty is working on a branch of the MySQL code base, called **MariaDB**.

Related Posts

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

On Eventual Consistency– An interview with Justin Sheehy. by Roberto V. Zicari, August 15, 2012

MariaDB: the new MySQL? Interview with Michael Monty Widenius. by Roberto V. Zicari on September 29, 2011

##

From → uncategorized

2 Comments Leave one →

dwight permalink

It really depends on the use case.

For example you want strong consistency for these:
- new user registration, “pick your username”. you don’t want two users to end up with “joe123″ username ever.

Eventual consistency use case:
- logging. i want the log even if not immediately readable and i may not want to fail just because of logging.

The other nuance with eventual consistency is the developer must keep the semantics in mind at all times. In QA you may not see any lag manifest so be sure to test the non-immediate case in QA when going that route.

Jonathan Gennick permalink

Monty’s comment about not handing out money when communications are down caught my eye. Believe it or not, once my local bank just handed out money. I live in a very small town. People tend to know each other. Shortly after moving here from a large city, I walked into my bank to withdraw some pocket money. I wanted $40, I think, or maybe $60. The teller told me their communications were down, and they couldn’t use their computers. She handed me the cash without making me fill out any forms, without even checking my identification. I’m not even sure she asked my name. (She must have recognized me). She did make a note so she could record the transaction later. I was pretty stunned at the time. Small-town living, I guess. Not sure they would do the same today, but twelve years ago it really happened. And yes, she did remember to record the transaction later.

Leave a Reply

Name (required)

Email (required, will not be published)

Website

Comment

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

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.

October 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				

« Sep Nov »

Search

Search