HBase

A Comprehensive Introduction

James Chin, Zikai Wang
Monday, March 14, 2011
CS 227 (Topics in Database Management)
CIT 367

Overview

Overview: History

- Began as project by Powerset to **process massive amounts of data** for natural language search

- Open-source implementation of Google’s **BigTable**
  - Lots of **semi-structured data**
  - Commodity Hardware
  - Horizontal Scalability
  - Tight integration with **MapReduce**

- Developed as part of Apache’s **Hadoop** project and runs on top of **HDFS (Hadoop Distributed Filesystem)**
  - Provides **fault-tolerant** way of storing **large quantities of sparse data**.

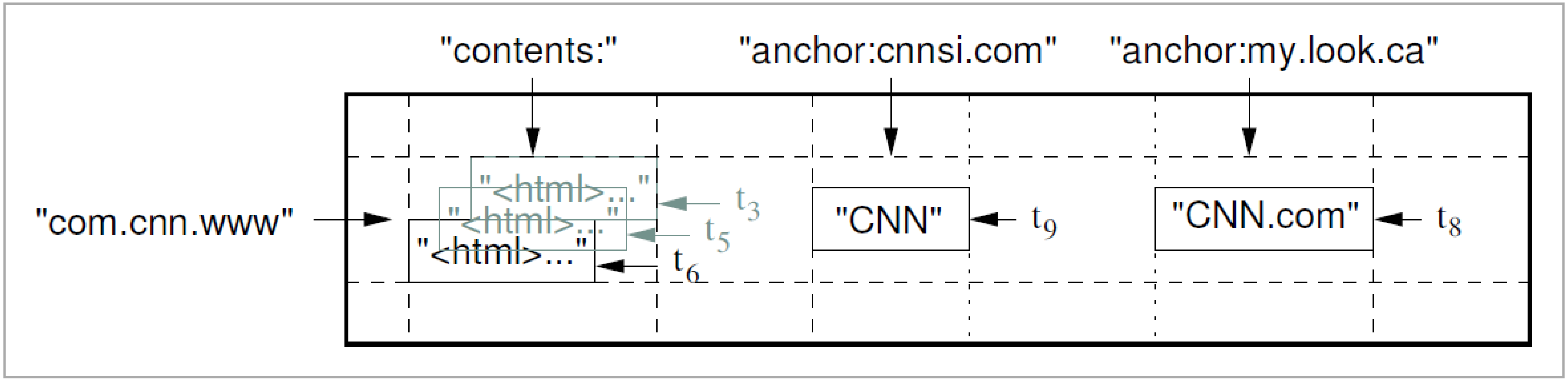
Overview: What is HBase?

- Non-relational, distributed database
- Column-Oriented
- Multi-Dimensional
- High Availability
- High Performance

Data Model & Operators

Data Model

- A **sparse**, **multi-dimensional**, **sorted** map
  - {row, column, timestamp} -> cell
- Column = **Column Family** : Column Qualifier



- Rows are **sorted lexicographically** based on row key
- **Region:** contiguous set of sorted rows
- HBase: a large number of columns, a low number of column families (2-3)

Operators

- Operations are based on **row keys**

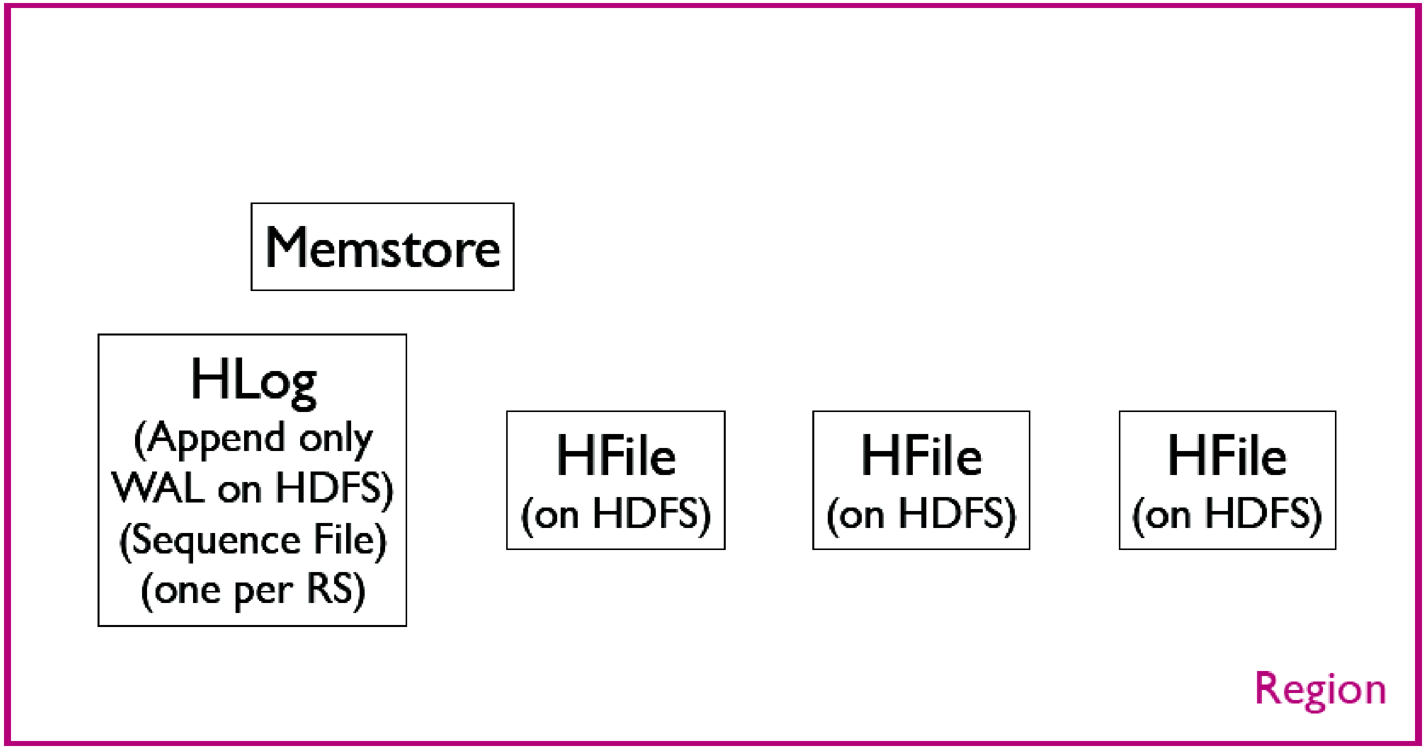
- **Single-row operations:**
  - Put
  - Get
  - Scan

- **Multi-row operations:**
  - Scan
  - MultiPut

- No built-in joins (use MapReduce)

Physical Structures

Physical Structures: Data Organization



- **Region:** unit of distribution and availability
- Regions are split when grown too large
- Max region size is a tuning parameter
    - Too low: prevents parallel scalability
    - Too high: makes things slow

Physical Structures: Need for Indexes

- HBase has **no built-in support for secondary indexes**
- API only exposes operations by **row key**

Row Key	Name	Position	Nationality
“1”	Nowitzki, Dirk	PF	Germany
“2”	Kaman, Chris	C	Germany
“3”	Gasol, Paul	PF	Spain
“4”	Fernandez, Rudy	SG	Spain

- **Find all players from Spain?**
  - With built-in API, scan the entire table
  - Manually build a secondary index table
  - Exploit the fact that rows are sorted lexicographically by row key based on byte order

Physical Structures: Secondary Index

- Data Table:

Row Key	Name	Position	Nationality
“1”	Nowitzki, Dirk	PF	Germany
“2”	Kaman, Chris	C	Germany
“3”	Gasol, Paul	PF	Spain
“4”	Fernandez, Rudy	SG	Spain

- **Index table on nationality column**
  - a scan operation
  - start row = "Spain"
  - stop scanning: set a RowFilter with a BinaryPrefixComparator on the end value("Spain")
  - range queries are also supported

Row Key	Dummy
“Germany 1”	Germany 1
“Germany 2”	Germany 2
“Spain 3”	Spain 3
“Spain 4”	Spain 4

Physical Structures: Secondary Index (cont.)

- **Find all power forwards from Spain?**
  - A composite index

- Row keys are **plain byte arrays**
  - Byte order = your desired order?
  - Convert strings, integers, floats, decimals carefully to bytes
  - Default sorting is ascending; if descending indexes are needed, reverse bit order

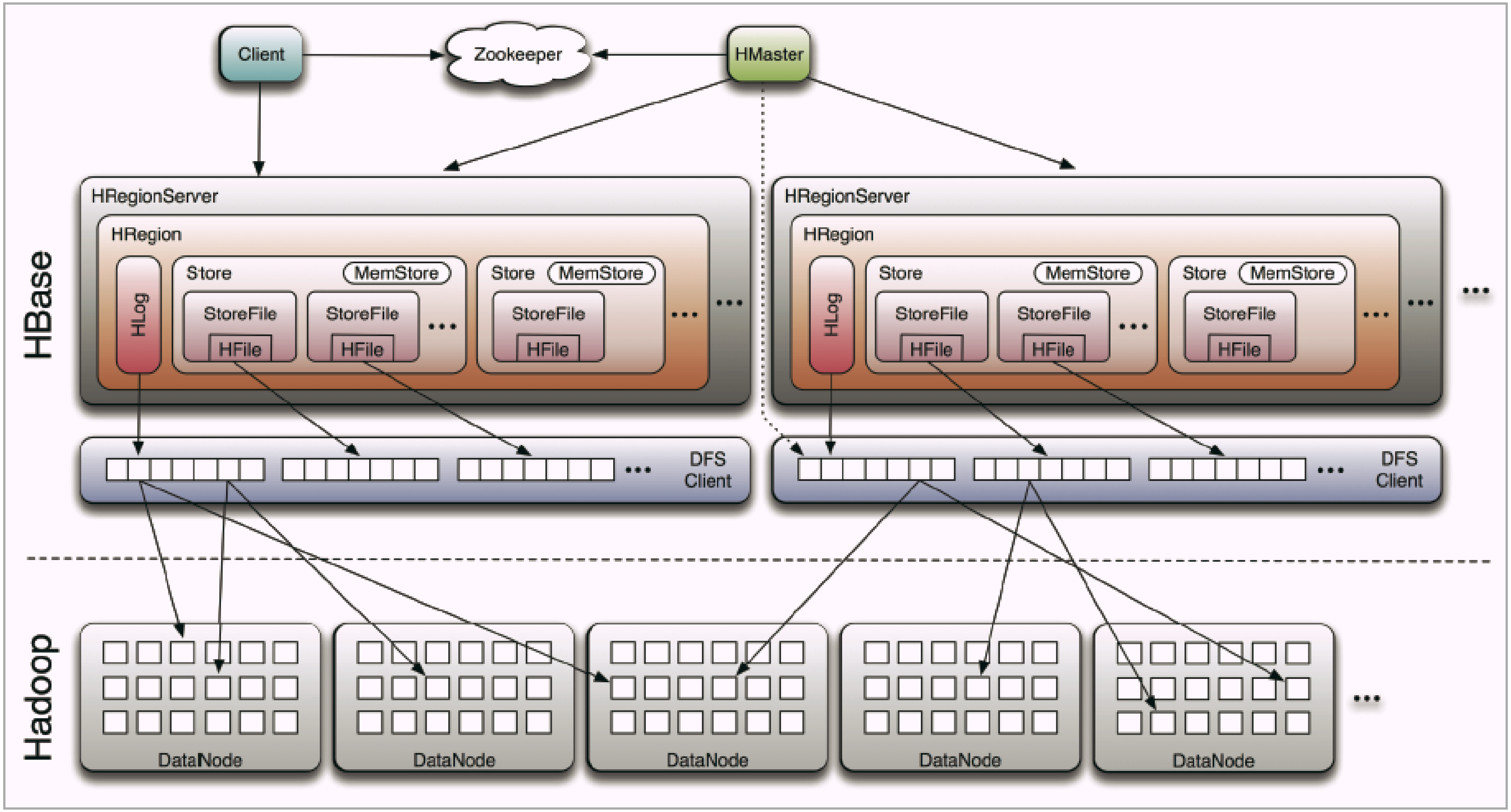
Physical Structures: More Indexing

- **Lily's** HBase Indexing Library
  - Aids in building and querying indexes in HBase
  - Hides the details of playing with byte[] row keys

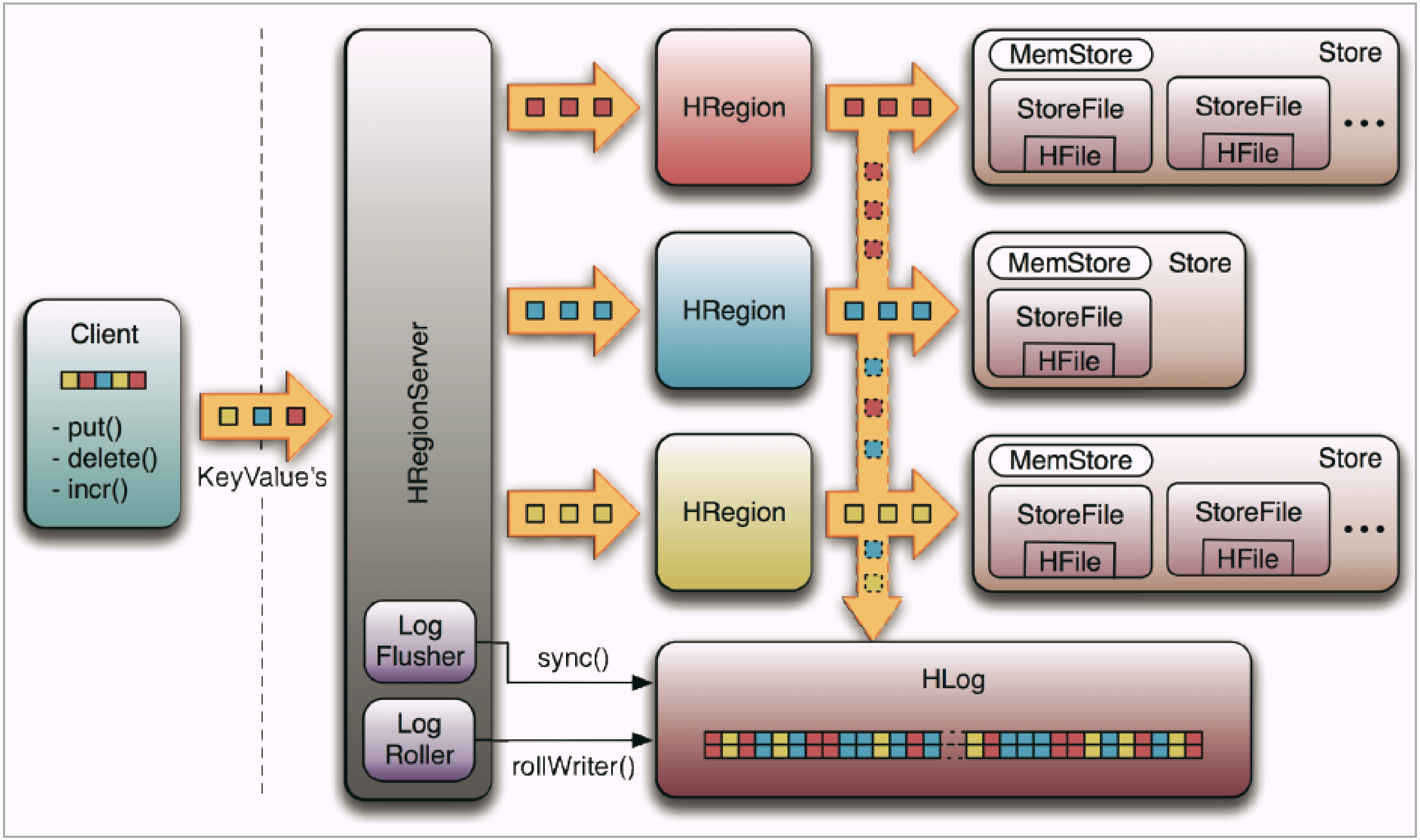
- **HBase + full text indexing and searching systems**
  - Apache Lucene (Apache Solr, elasticsearch)
  - Lily, HAvroBase (HBase + Solr), HBasene (HBase + Lucene)

System Architecture

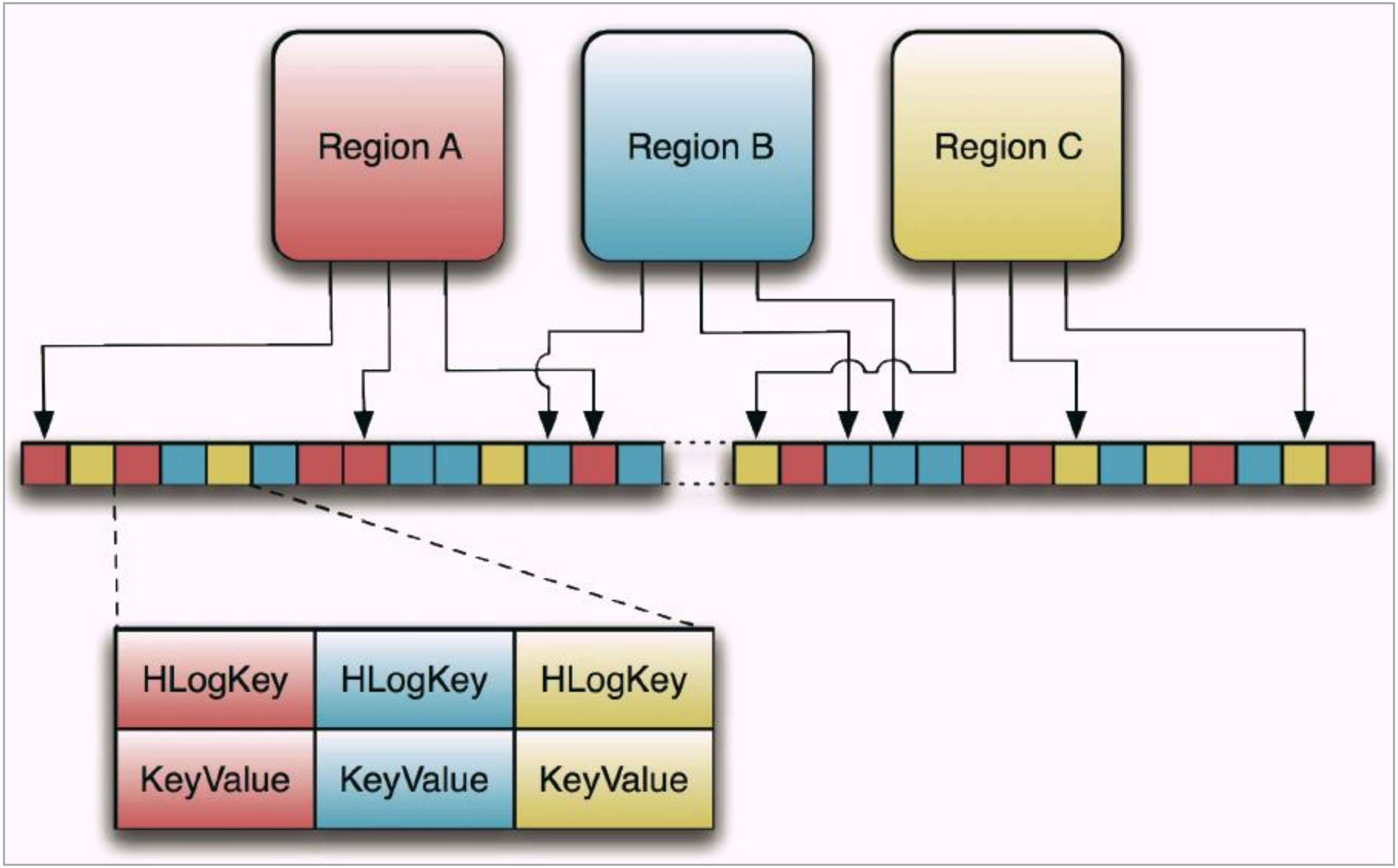
System Architecture: Overview



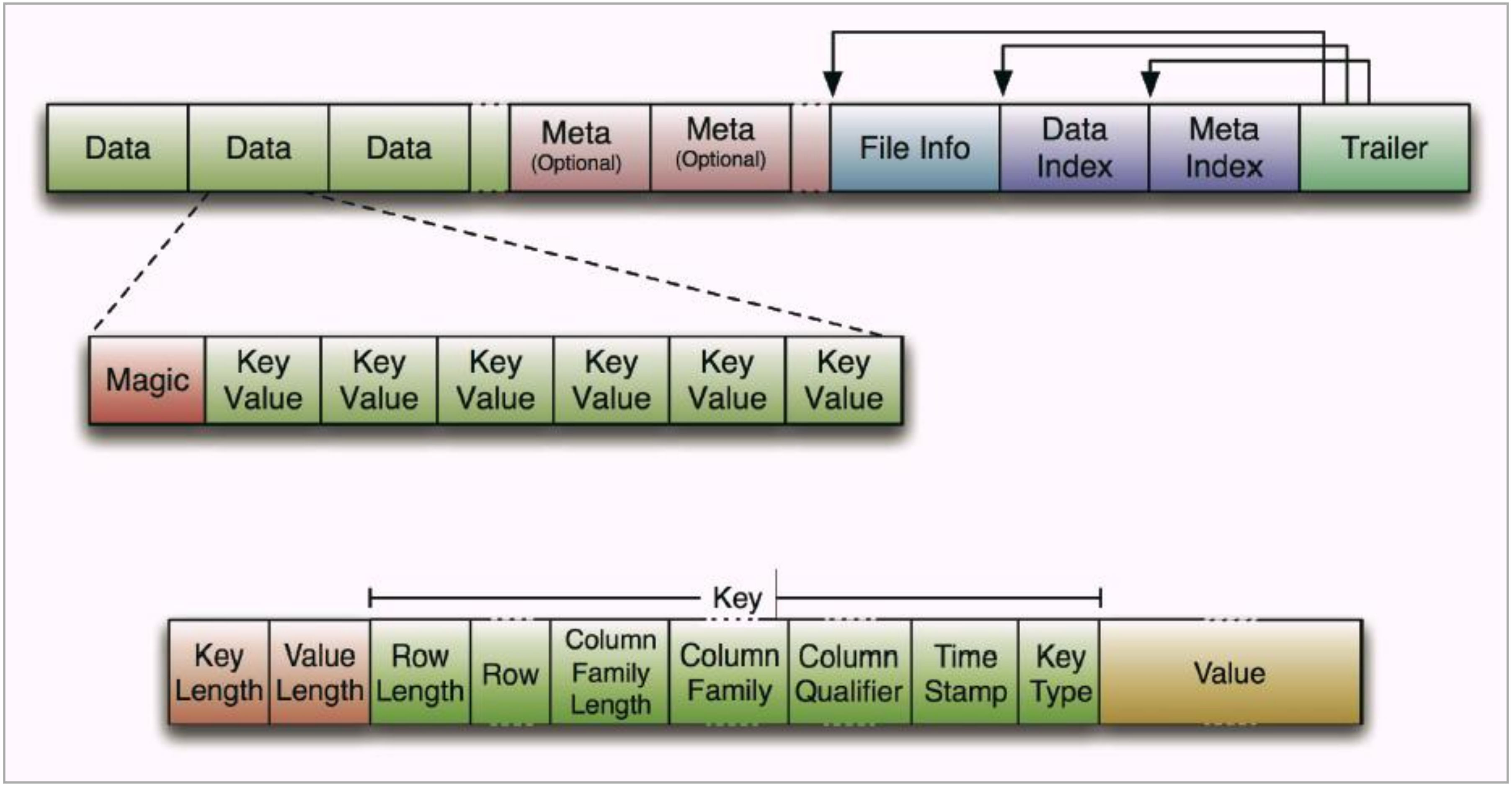
System Architecture: Write-Ahead-Log Flow



System Architecture: WAL (cont.)



System Architecture: HFile and KeyValue



APIs

APIs: Overview

- **Java**
  - Get, Put, Delete, Scan
  - IncrementColumnValue
  - TableInputFormat - MapReduce Source
  - TableOutputFormat - MapReduce Sink
- Rest
- Thrift
- Scala
- Jython
- Groovy DSL
- Ruby shell
- Java MR, Cascading, Pig, Hive

ACID Properties

ACID Properties

- HBase **not ACID-compliant**, but does guarantee certain specific properties

- **Atomicity**
  - All mutations are atomic within a row. Any put will either wholely succeed or wholely fail.
  - APIs that mutate several rows will *not* be atomic across the multiple rows.
  - The order of mutations is seen to happen in a well-defined order for each row, with no interleaving.

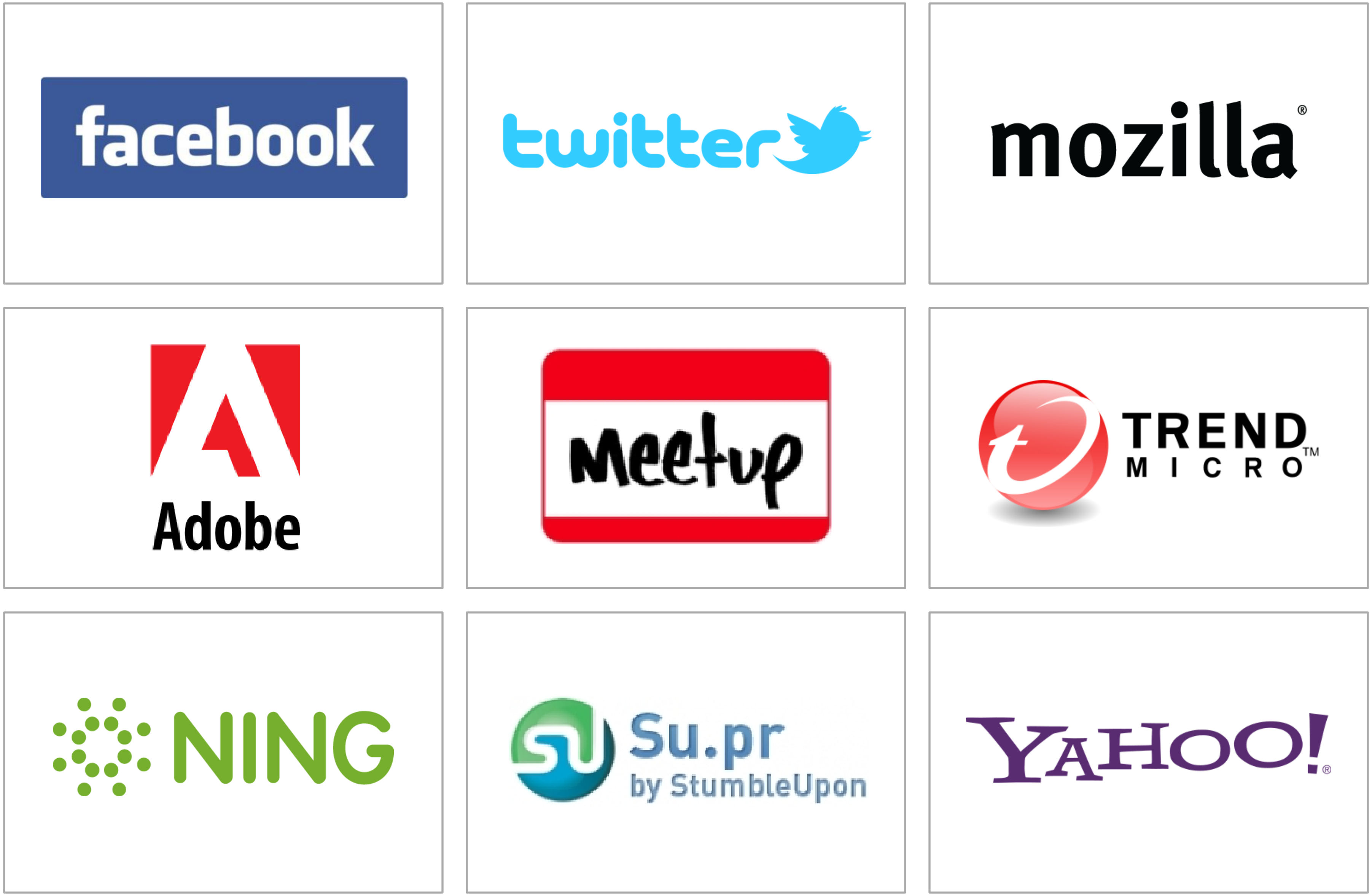
- **Consistency** and **Isolation**
  - All rows returned via any access API will consist of a complete row that existed at some point in the table's history.

ACID Properties (cont.)

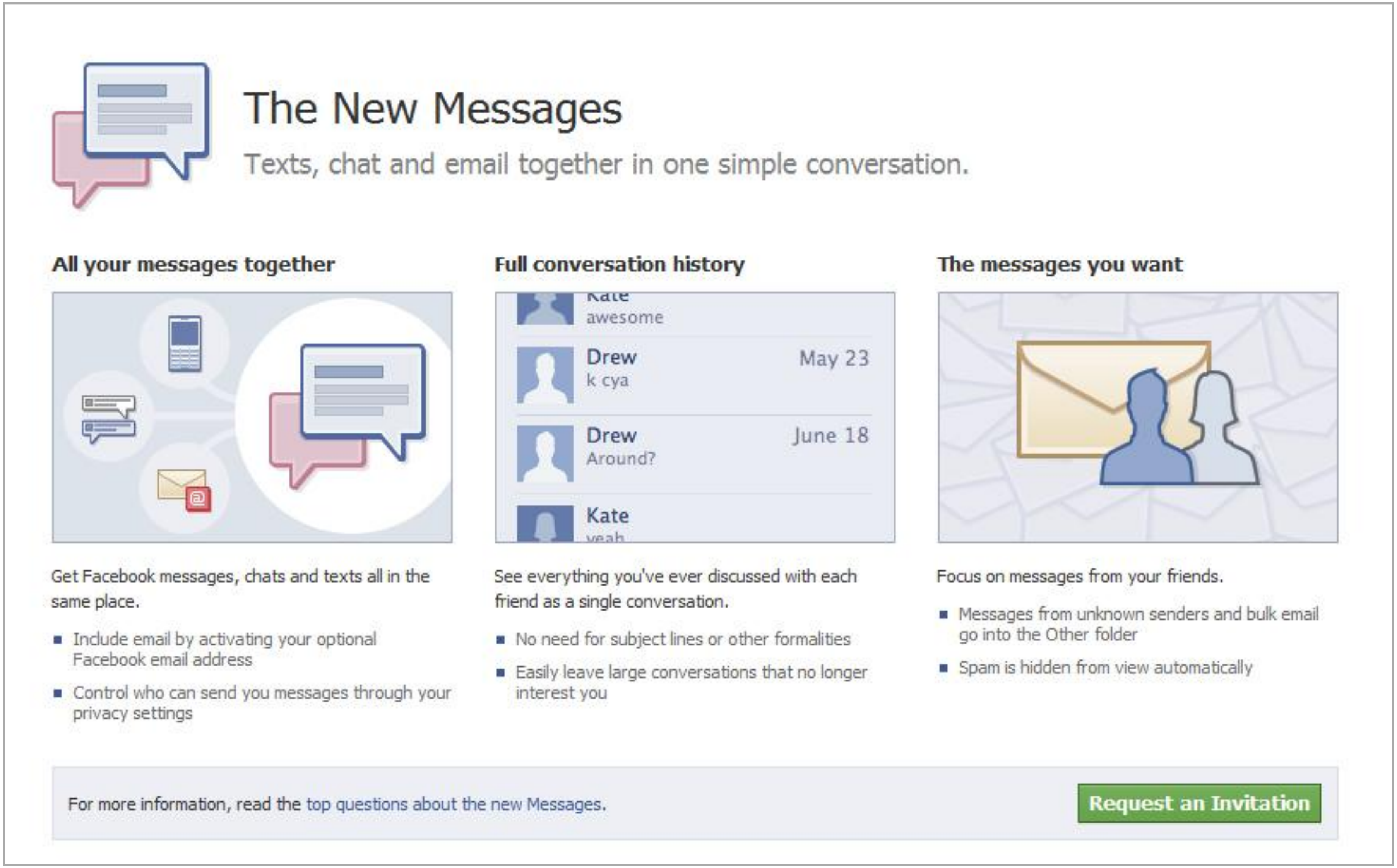
- **Consistency** of Scans
  - A scan is not a consistent view of a table. Scans do not exhibit snapshot isolation.
  - Those familiar with relational databases will recognize this isolation level as "read committed".
- **Durability**
  - All visible data is also durable data. That is to say, a read will never return data that has not been made durable on disk.
  - Any operation that returns a "success" code (e.g. does not throw an exception) will be made durable.
  - Any operation that returns a "failure" code will not be made durable (subject to the Atomicity guarantees above).
  - All reasonable failure scenarios will not affect any of the listed ACID guarantees.

Users

Users: Just to name a few…



Users: Facebook - Messaging System



The New Messages

Texts, chat and email together in one simple conversation.

All your messages together

Get Facebook messages, chats and texts all in the same place.

- Include email by activating your optional Facebook email address
- Control who can send you messages through your privacy settings

Full conversation history

Kate
awesome

Drew — May 23
k cya

Drew — June 18
Around?

Kate
yeah

See everything you've ever discussed with each friend as a single conversation.

- No need for subject lines or other formalities
- Easily leave large conversations that no longer interest you

The messages you want

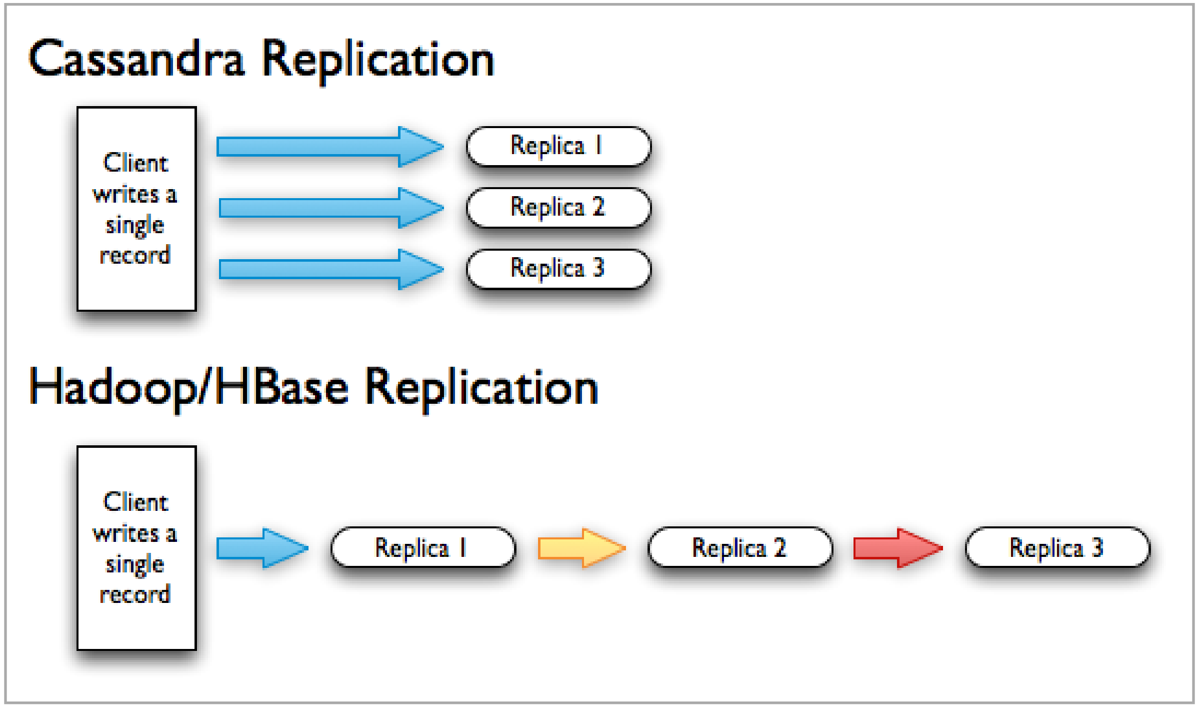
Focus on messages from your friends.

- Messages from unknown senders and bulk email go into the Other folder
- Spam is hidden from view automatically

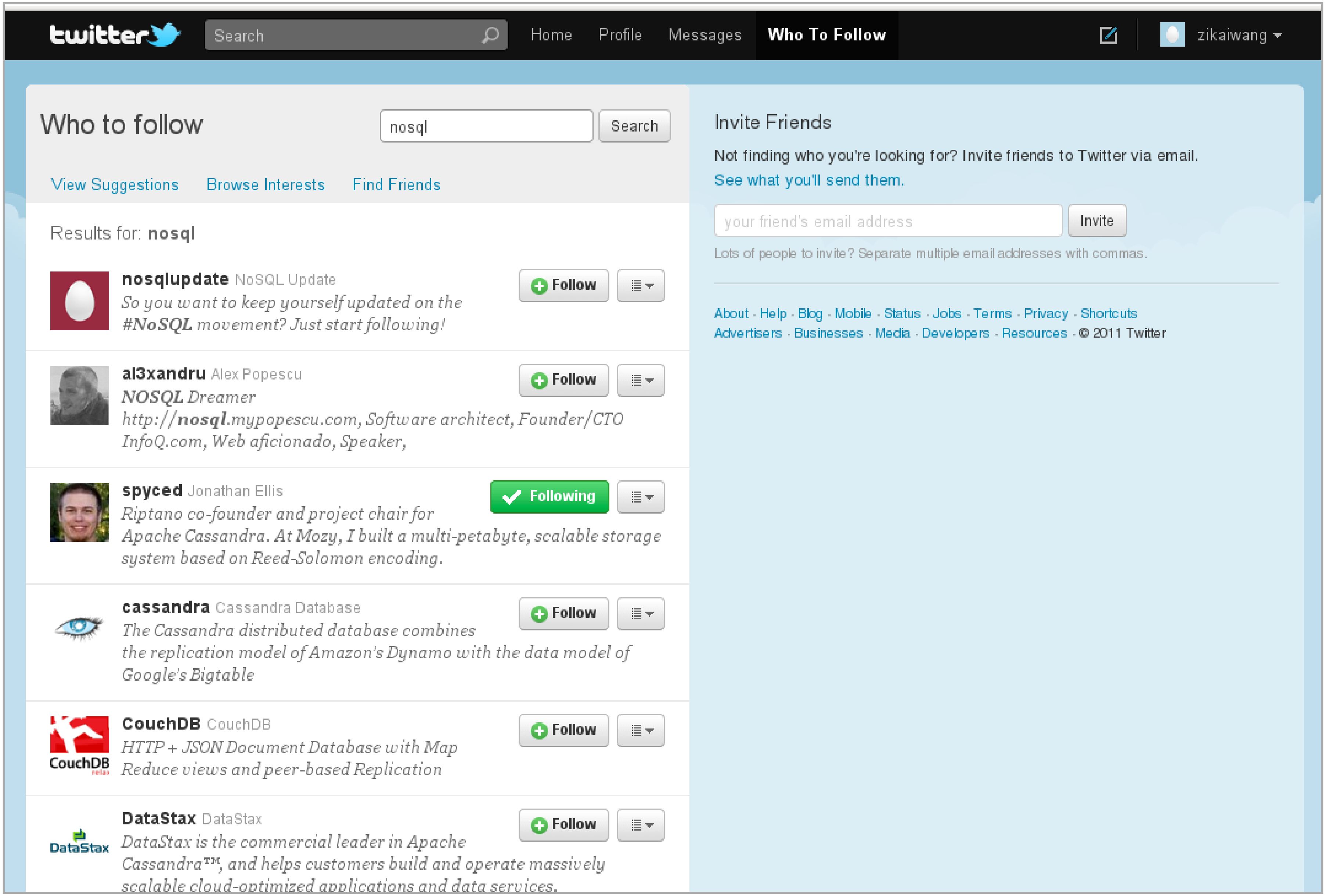
For more information, read the top questions about the new Messages.

Users: Facebook - Messaging System (cont.)

- **Previous Solution:** Cassandra
- **Current Solution:** HBase
- **Why?** Cassandra's replication behavior



Users: Twitter - People Search



Who to follow

nosql | Search

View Suggestions · Browse Interests · Find Friends

Results for: **nosql**

nosqlupdate NoSQL Update — Follow
So you want to keep yourself updated on the #NoSQL movement? Just start following!

al3xandru Alex Popescu — Follow
*NOSQL Dreamer http://**nosql**.mypopescu.com, Software architect, Founder/CTO InfoQ.com, Web aficionado, Speaker,*

spyced Jonathan Ellis — ✓ Following
Riptano co-founder and project chair for Apache Cassandra. At Mozy, I built a multi-petabyte, scalable storage system based on Reed-Solomon encoding.

cassandra Cassandra Database — Follow
The Cassandra distributed database combines the replication model of Amazon's Dynamo with the data model of Google's Bigtable

CouchDB CouchDB — Follow
HTTP + JSON Document Database with Map Reduce views and peer-based Replication

DataStax DataStax — Follow
DataStax is the commercial leader in Apache Cassandra™, and helps customers build and operate massively scalable cloud-optimized applications and data services.

Invite Friends

Not finding who you're looking for? Invite friends to Twitter via email.
See what you'll send them.

your friend's email address | Invite

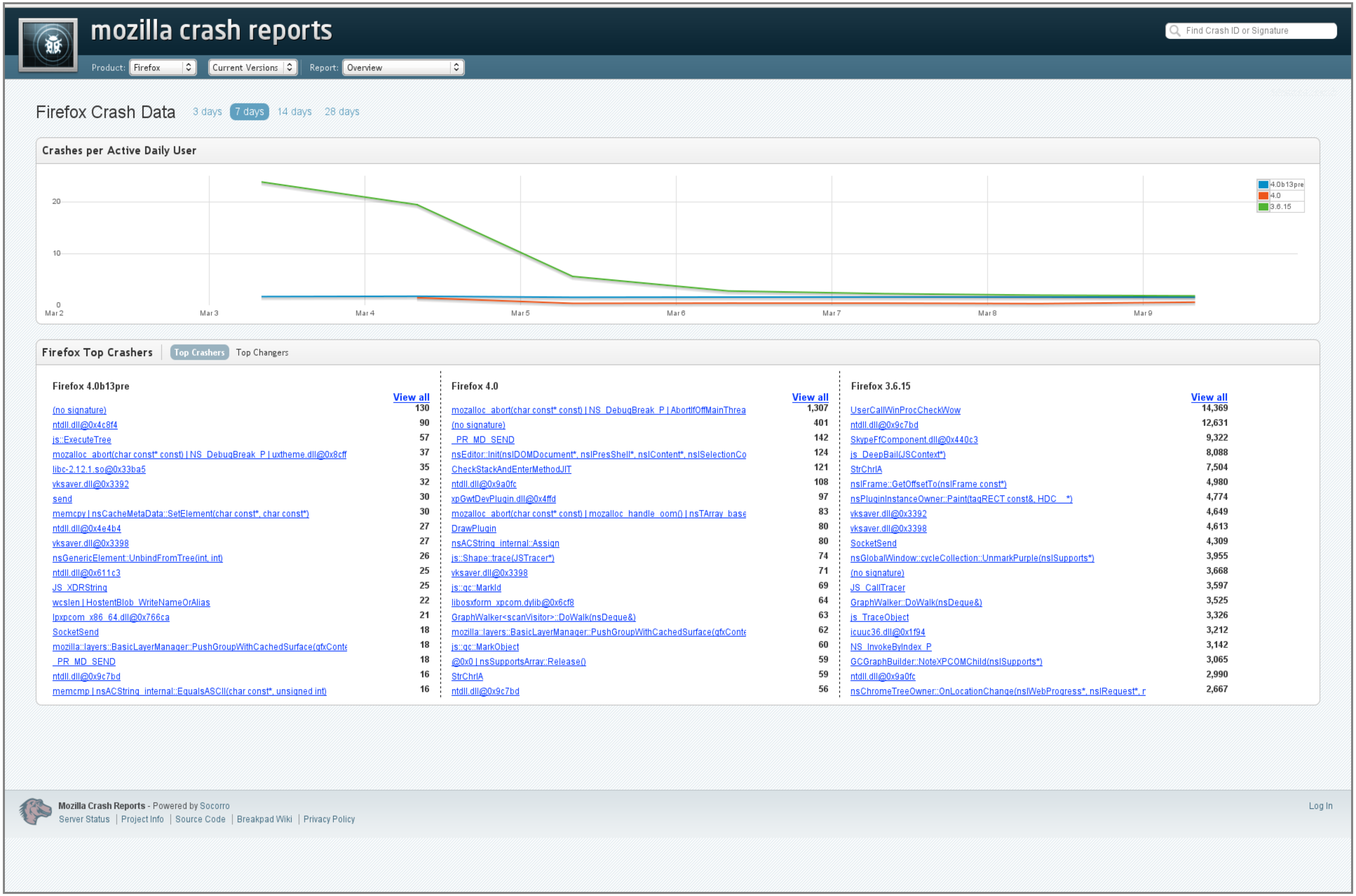
Lots of people to invite? Separate multiple email addresses with commas.

About · Help · Blog · Mobile · Status · Jobs · Terms · Privacy · Shortcuts
Advertisers · Businesses · Media · Developers · Resources · © 2011 Twitter

Users: Twitter - People Search (cont.)

- Customer Indexing
- **Previous Solution:** offline process at a single node
- **Current Solution:**
  - Import user data into HBase
  - Periodically MapReduce job reading from HBase
  - Hits FlockDB and other internal services in mapper
  - Write data to sharded, replicated, horizontally scalable, in-memory, low-latency Scala service
- **Vs. Others:**
  - HDFS: Data is mutable
  - Cassandra: OLTP vs. OLAP?

Users: Mozilla - Socorro



mozilla crash reports

Find Crash ID or Signature

Product: Firefox | Current Versions | Report: Overview

Firefox Crash Data

3 days | 7 days | 14 days | 28 days

Crashes per Active Daily User

Firefox Top Crashers

Top Crashers | Top Changers

Firefox 4.0b13pre	View all
(no signature)	130
ntdll.dll@0x4c8f4	90
js::ExecuteTree	57
mozalloc_abort(char const* const) | NS_DebugBreak_P | uxtheme.dll@0x8cff	37
libc-2.12.1.so@0x33ba5	35
vksaver.dll@0x3392	32
send	30
memcpy | nsCacheMetaData::SetElement(char const*, char const*)	30
ntdll.dll@0x4e4b4	27
vksaver.dll@0x3398	27
nsGenericElement::UnbindFromTree(int, int)	26
ntdll.dll@0x611c3	25
JS_XDRString	25
wcslen | HostentBlob_WriteNameOrAlias	22
lpxpcom_x86_64.dll@0x766ca	21
SocketSend	18
mozilla::layers::BasicLayerManager::PushGroupWithCachedSurface(gfxConte	18
_PR_MD_SEND	18
ntdll.dll@0x9c7bd	16
memcmp | nsACString_internal::EqualsASCII(char const*, unsigned int)	16

Firefox 4.0	View all
mozalloc_abort(char const* const) | NS_DebugBreak_P | AbortIfOffMainThrea	1,307
(no signature)	401
_PR_MD_SEND	142
nsEditor::Init(nsIDOMDocument*, nsIPresShell*, nsIContent*, nsISelectionCo	124
CheckStackAndEnterMethodJIT	121
ntdll.dll@0x9a0fc	108
xpGwtDevPlugin.dll@0x4ffd	97
mozalloc_abort(char const* const) | mozalloc_handle_oom() | nsTArray_base	83
DrawPlugin	80
nsACString_internal::Assign	80
js::Shape::trace(JSTracer*)	74
vksaver.dll@0x3398	71
js::gc::MarkId	69
libosxform_xpcom.dylib@0x6cf8	64
GraphWalker<scanVisitor>::DoWalk(nsDeque&)	63
mozilla::layers::BasicLayerManager::PushGroupWithCachedSurface(gfxConte	62
js::gc::MarkObject	60
@0x0 | nsSupportsArray::Release()	59
StrChrIA	59
ntdll.dll@0x9c7bd	56

Firefox 3.6.15	View all
UserCallWinProcCheckWow	14,369
ntdll.dll@0x9c7bd	12,631
SkypeFfComponent.dll@0x440c3	9,322
js_DeepBail(JSContext*)	8,088
StrChrIA	7,504
nsIFrame::GetOffsetTo(nsIFrame const*)	4,980
nsPluginInstanceOwner::Paint(tagRECT const&, HDC__*)	4,774
vksaver.dll@0x3392	4,649
vksaver.dll@0x3398	4,613
SocketSend	4,309
nsGlobalWindow::cycleCollection::UnmarkPurple(nsISupports*)	3,955
(no signature)	3,668
JS_CallTracer	3,597
GraphWalker::DoWalk(nsDeque&)	3,525
js_TraceObject	3,326
icuuc36.dll@0x1f94	3,212
NS_InvokeByIndex_P	3,142
GCGraphBuilder::NoteXPCOMChild(nsISupports*)	3,065
ntdll.dll@0x9a0fc	2,990
nsChromeTreeOwner::OnLocationChange(nsIWebProgress*, nsIRequest*, r	2,667

Mozilla Crash Reports - Powered by Socorro

Server Status | Project Info | Source Code | Breakpad Wiki | Privacy Policy

Log In

Users: Mozilla – Socorro (cont.)

- **Socorro**, Mozilla’s crash reporting system (https://crash-stats.mozilla.com/products)
  - Catches, processes, and presents crash data for Firefox, Thunderbird, Fennec, Camino, and Seamonkey.
- 2.5 million crash reports per week, 320GB per day
- **Previous Solution:** NFS (raw data), PostgreSQL (analyze results)
  - 15% of crash reports are processed
- **Current Solution:** Hadoop (processing) + HBase (storage)

HBase vs. RDBMS

HBase vs. RDBMS

HBase	RDBMS
Column-oriented	Row oriented (mostly)
Flexible schema, add columns on the fly	Fixed schema
Good with sparse tables	Not optimized for sparse tables
No query language	SQL
Wide tables	Narrow tables
Joins using MR – not optimized	Optimized for joins (small, fast ones too!)
Tight integration with MR	Not really...

HBase vs. RDBMS (cont.)

HBase	RDBMS
De-normalize your data	Normalize as you can
Horizontal scalability – just add hardware	Hard to shard and scale
Consistent	Consistent
No transactions	Transactional
Good for semi-structured data as well as structured data	Good for structured data

Questions?

Thanks!