

Google™

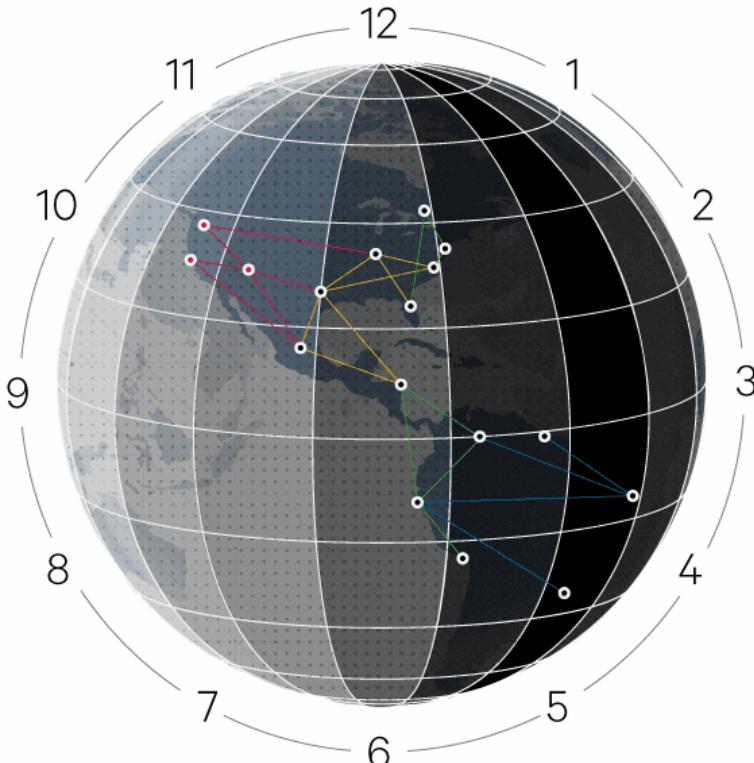
# Spanner

Google's Distributed Database  
Sebastian Kanthak

# Google Spans Entire Planet With GPS-Powered Database

BY CADE METZ 09.19.12 6:30 AM

[Follow @cademetz](#)



Three years ago, a top Google engineer named Vijay Gill was asked what he would do if someone gave him a magic wand.

## Dropbox for Business

[Click for a free trial](#)



### MOST RECENT WIRED POSTS



Opinion: Failure of Neal Stephenson Kickstarter Is Apparently Everyone's Fault But the Developer's



RSA Tells Its Developer Customers: Stop Using NSA-Linked Algorithm



Enjoy Heavenly Iced Coffee at Home With This Slow-Drip Cold Brew System



Hiroshi Yamauchi, Man Who Built Nintendo's Gaming Empire, Dies at 90

# Google's Spanner: Database Tech That Can Scan the Planet

# 1.3k

SHARES

120

1.1k

108

59

0

4



Ads by Google

**Dashboard Best Practices** - IT Manager's Guide to Dashboards Download Your Free White Paper Now!

[logianalytics.com/Dashboard-WP](http://logianalytics.com/Dashboard-WP)

**2013's 10 Best Web Hosts** - Our Best Hosts of 2013 - Revealed. Read This Before Buying.

[www.consumer-rankings.com/Hosting](http://www.consumer-rankings.com/Hosting)

**Kimball Group White Paper** - Essential Steps for Integrated EDW. Download Free White Paper Now!

[www.informatica.com](http://www.informatica.com)



## What's Hot



US &amp; WORLD

How Much Longer Can Earth Support Life?

691 SHARES



SOCIAL MEDIA

Apple Users Sent 7 Million iOS 7 Tweets in Two Days

1.3K SHARES

#1 YOUTUBE VIDEO PROMOTION SERVICE

# •••{devbeat}•••

November 12+13;  
The Regency Center;  
San Francisco;

## Google's scary powerful Spanner database gets us one step closer to Skynet



November 26, 2012 4:01 PM

Sean Ludwig



[Like](#) 232 [g+1](#) 14 [Tweet](#) 139 [Share](#) [VB More](#)

3 Comments

Google's [Spanner](#) is a single database that runs across hundreds of data centers throughout the world. It's so smart that it rapidly shifts resources during outages without human intervention and keeps everything

### DataBeat 2013

Dec. 4 - 5, 2013  
Redwood City, CA

Early Bird Tickets on Sale

 New Relic®

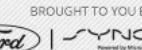
Win the Nerd Lottery!

Get the ULTIMATE DEVELOPER BUNDLE!

Deploy Now!

Did you miss **MobileBeat** 2013?

Click here for all of the news and announcements



white papers

[View more](#)

EMAIL  
MARKETING  
SIMPLIFIED



emma

[Advertise on VentureBeat](#)

# Motivation

Heritage: BigTable, Megastore

Spanner Features:

- Familiar row/column data model
- Synchronous Replication
- Transactions across rows



# System Overview

# Logical Data Layout

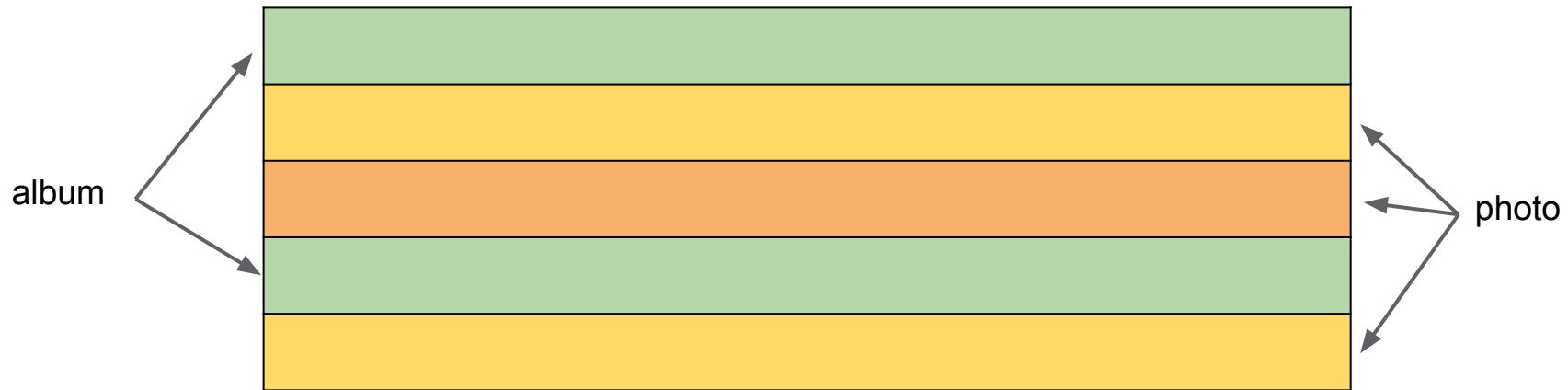
Albums

user_id	album_id	name
1	1	Maui
1	2	St. Louis

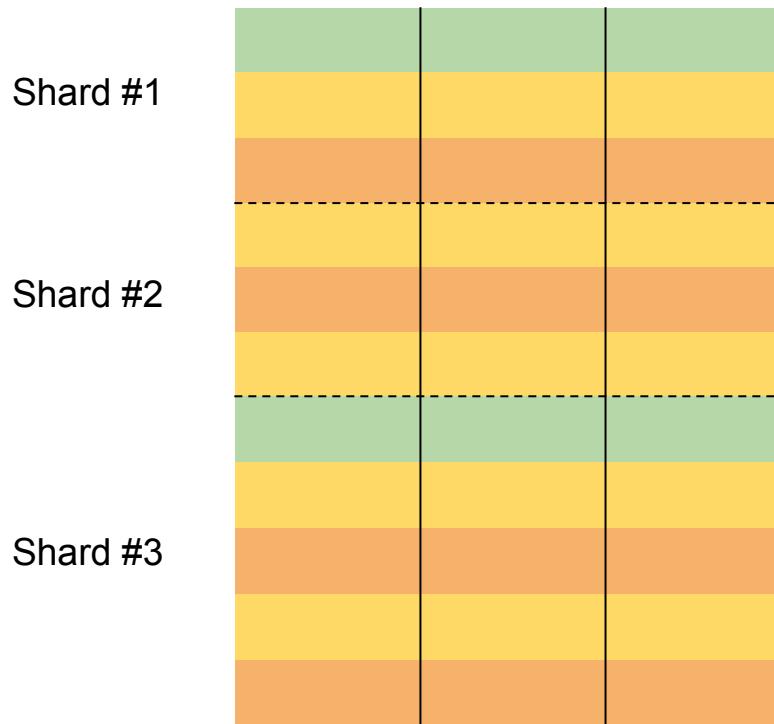
Photos

user_id	album_id	photo_id	title
1	1	2	Beach
1	1	5	Snorkeling
1	2	3	Gateway Arch

# Physical Data Layout: Interleaved Tables



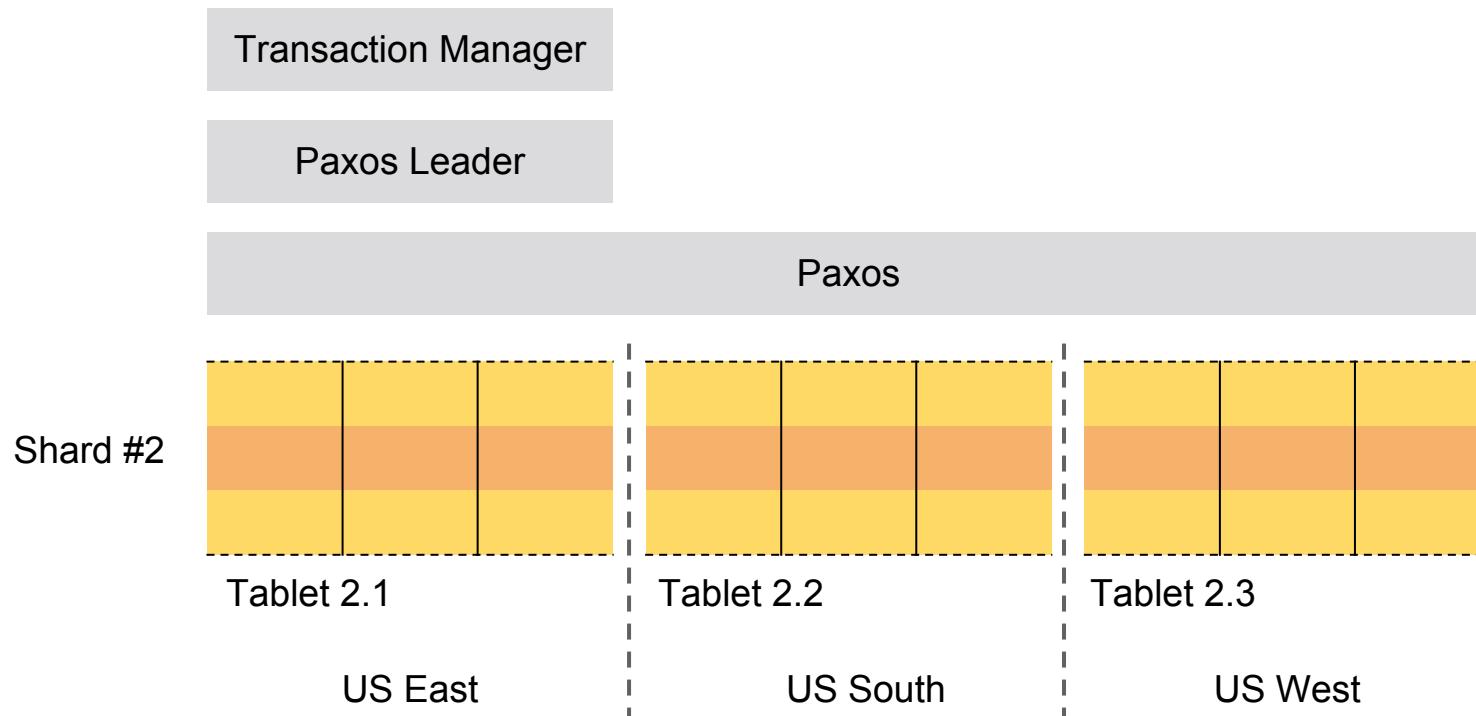
# Sharding



But still support:

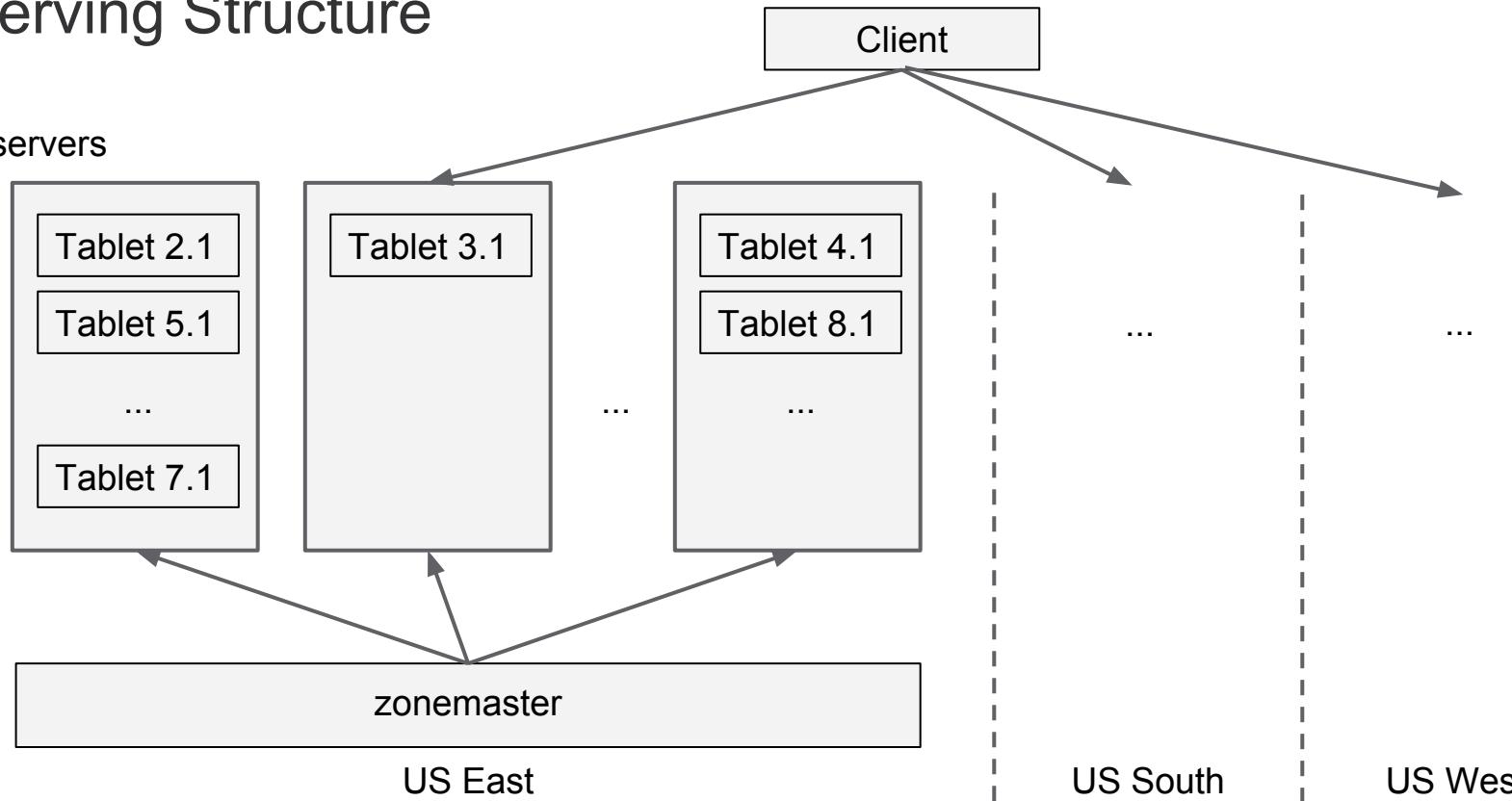
- Transactions across shards
- Consistent snapshot reads  
(range scans) across shards

# Replication



# Serving Structure

spanservers

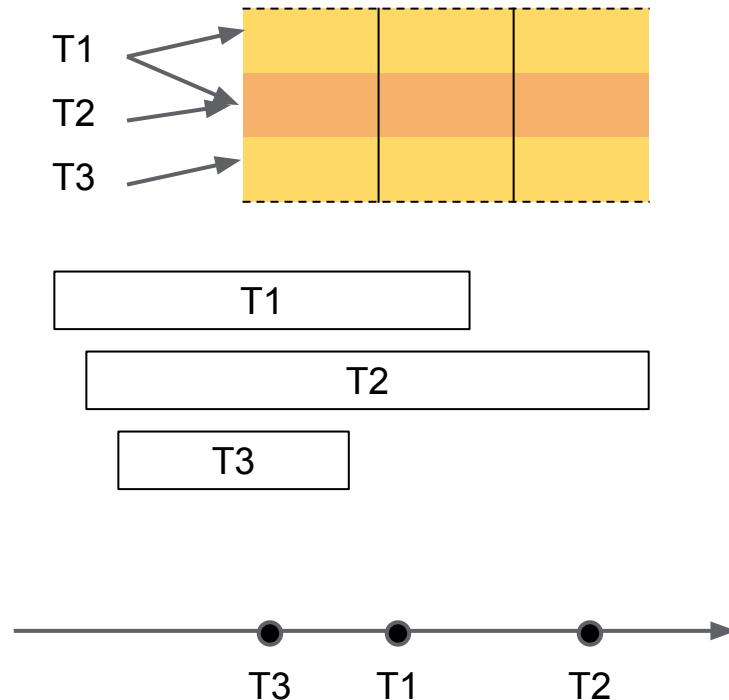


# Transactions & Concurrency

# Strict Two-Phase Locking

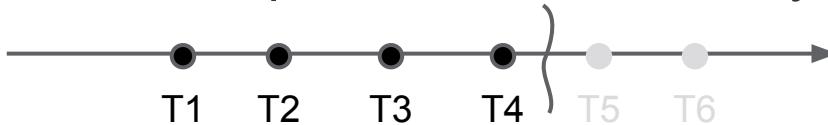
Life of a transaction:

1. Acquire locks
2. Execute reads
3. Pick commit timestamp
4. Replicate writes (through paxos)
5. Ack commit
6. Apply writes
7. Release locks



# Snapshot Reads

Choose a prefix of commit history



Properties of snapshots:

- immutable
- consistent

Can be used for:

- long-running batch operations (e.g. map reduce)
- stale reads (e.g. 10s old)
- strong (current) reads: lock-free, don't block writers

# Picking commit timestamps

Attempt #1: Assign from local (monotonic) clock

1. Acquire locks
2. Execute reads
3. **Pick commit timestamp = now()**
4. Replicate writes (through paxos)
5. Ack commit
6. Apply writes
7. Release locks



# Example: Ad System

Campaigns

campgain_id	keyword	bid
4	strange loop	\$2.00

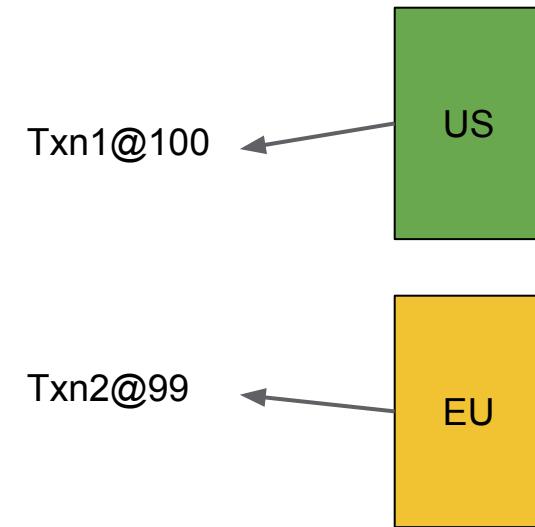
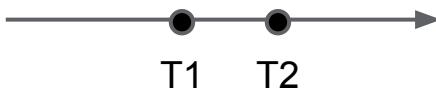
- On US server
- On EU server

Impressions

region	time	campaign_id	cost
US	2013/09/20-07...	4	\$1.50
US			
...			
EU	2013/09/20-06...	4	\$0.50
EU			

## Example: What goes wrong

- Txn 1 creates a new ad on US server
- Ad serving system notified
- Ad server in Europe
- User clicks on ad
- Txn 2 logs click on EU server



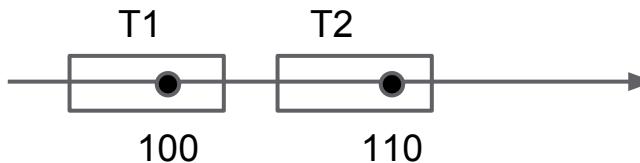
Invariant: Any snapshot that contains txn 2 should also contain txn 1.

# External Consistency

Definition:

If T1 commits before T2 starts, T1 should be serialized before T2.  
In other words, T2's commit timestamp should be greater than T1's commit timestamp.

Note: Applies even if T1 and T2 don't conflict.



# TrueTime

Idea: There is a global “true” time  $t$

$\text{TT.now()} = [\text{earliest}, \text{latest}] \ni t.$

- $\text{TT.now().earliest}$  definitely in the past
- $\text{TT.now().latest}$  definitely in the future



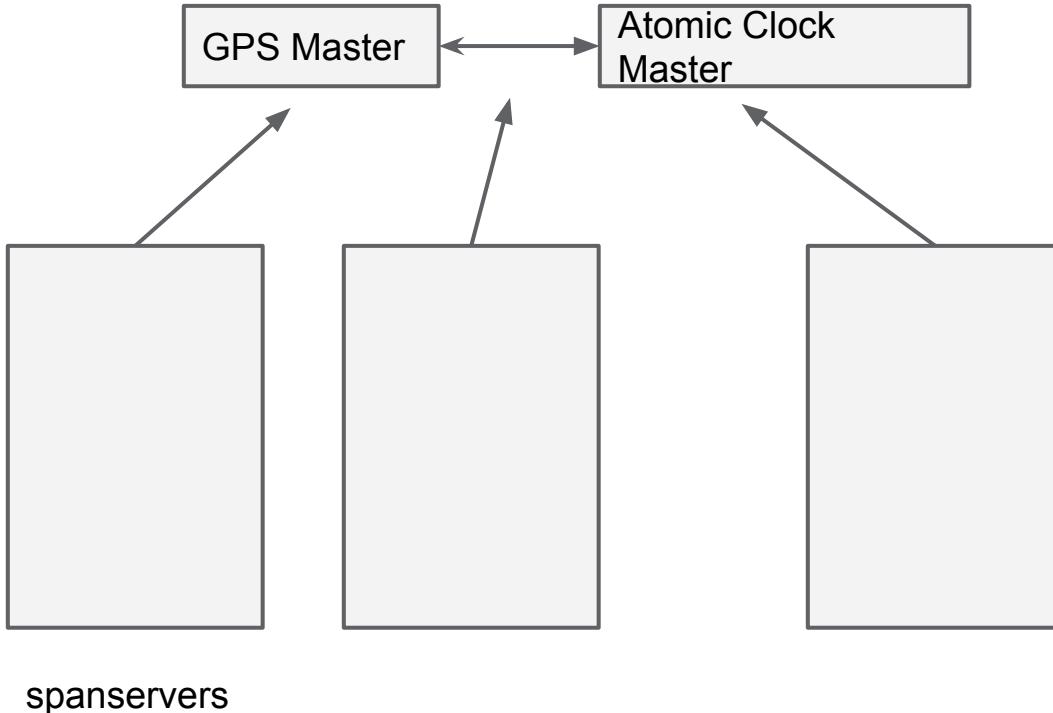
## Timestamp assignment: TrueTime

Transaction protocol becomes:

1. Acquire locks
2. Execute reads
- 3. Pick commit timestamp  $T = TT.now().latest$**
4. Replicate writes (through paxos)
- 5. Wait until  $TT.now().earliest > T$**
6. Ack transaction commit
7. Apply write
8. Release locks

Strong reads:  $T = TT.now().latest$

# True Time: Architecture



periodic poll: [earliest, latest]

In-between polls, uncertainty radius grows based on worst-case clock drift (200 usec / sec)

# Conclusions

Don't sacrifice semantics

Move complexity lower in the stack

Make clock uncertainty explicit: Known unknowns are better than unknown unknowns

Thank you!

Questions?

Sebastian Kanthak  
skanthak@google.com