Newer Post Slider(Newer 20) Home Random Post Slider(Random 20) Slider(Older 20) Older Post

Scalability Rules: 50 Principles for Scaling Web Sites

Scalability Rules: 50 Principles for Scaling Web Sites
9. Design for Fault Tolerance and Graceful Failure
Not designed to fail.

Rule 36—Design Using Fault Isolative “Swimlanes” ==> todo
Shard: split data

Rule 37—Never Trust Single Points of Failure
Identify single instances on architectural diagrams. Strive for active/active configurations.
Strive for active/active rather than active/passive solutions. Use load balancers to balance traffic across instances of a service. Use control services with active/passive instances for patterns that require singletons.

Rule 38—Avoid Putting Systems in Series
Components in series have a multiplicative effect of failure.
When necessary to do so add multiple versions of that component so that if one fails others are available to take its place.

Rule 7—Design to Clone Things (X Axis)
Simply clone services and implement a load balancer.

For databases, ensure the accessing code understands the difference between a read and a write.

Just because the data is out of sync with the primary transactional database by 3 or 30 or 90 seconds doesn’t mean that it isn’t correct, just that there is a chance that it isn’t correct.

MySQL implements replication through the master-slave concept—the master database being the primary transactional database that gets written to and the slave databases are read-only copies of the master databases.

Rule 8—Design to Split Different Things (Y Axis)
scaling through the separation of distinct and different functions and data, by either nouns or verbs or a combination of both nouns and verbs.
Separate sets of data, so we can scale them differently.
Scale code bases.

Rule 9—Design to Split Similar Things (Z Axis): sharding and podding
take one data set or service and partitioning it into several pieces.

Rule 15—Firewalls, Firewalls Everywhere!
Use firewalls only when they significantly reduce risk and recognize that they cause issues with scalability and availability.
Firewalls can lower availability and cause unnecessary scalability chokepoints.

Rule 16—Actively Use Log Files
Splunk, Cricket or Cacti
summarizing logs over time and archiving and purging them as their value decreases.
logging in an asynchronous fashion.

5. Don’t Duplicate Your Work
Rule 18—Stop Redirecting Traffic
A slightly better way to redirect with HTML is to use the meta tag “refresh” and automatically send the user’s browser to the new page.
<meta http-equiv="Refresh" content="0;url=http://www.akfpartners.com/techblog" />

Popular Posts

Design a chess game using OO principles | Runhe Tian Coding Practice

[CC150v5] 8.4 Design a Parking Lot - Shuatiblog.com

Avro vs Protocol Buffers vs Thrift

thought-works: Object Oriented design for Elevator in a multi-storied apartment

How to design a tiny URL or URL shortener? - GeeksforGeeks

Design Hit Counter - how to count number of requests in last second, minute and hour - Stack Overflow

Solr vs. Elasticsearch

Design a chat server | Hello World

Difference between System.exit() and System.halt() method?

Design an in-memory file system

Labels

- Algorithm (13)
- Architecture Principles (11)
- Big Data (24)
- Book Notes (57)
- Brain Teaser (12)
- Code Quality (10)
- Company - LinkedIn (11)
- Concurrency (10)
- Cracking Code Interview (22)
- Database (24)
- Design Patterns (29)
- Distributed (19)
- Google (11)
- How to (13)
- Interview (53)
- Interview - Review (21)
- Interview Q&A (20)
- Interview-Java (32)
- Interview-System Design (65)
- Java (93)
- Java - Code (21)
- Linux (35)
- Miscs (25)
- MultiThread (22)

Blog Archive

► 2016 (312)
▼ 2015 (724)
► Decemb
► Novemb
► October
► Septemb
► August (
► July (140
▼ June (39
Scalabilit Web S
AKF's Mo Archite
Pragmati Scalat
Design a
Finding a photo
How to In Using
POI-Geol
Design H of requ
Code fan imperi
Google A
Design R
Rejected Micros
Summary Miscs
Sticky Se my blo
How Twit Day U
Twitter di increm ITeye
System D Hashi
How to d shorte
Instagran Teraby
Implemer
Flickr Arc
The Codi
Designs, Buildir
Numbers Everyt
Fallacies Wikipe
How to A Palant
checkche
The Arch Applic
Why Strir
Design ic Dong'
Scalabilit
How to p in a te

- Operating System (13)
- Product Architecture (27)
- Security (11)
- System Design (156)
- System Design - Practice (19)
- Testing (10)
- to-do (40)
- Tools (10)

Pages

- Home
- System Design: Lesson Learned
- Video: System Desgin
- resources

we can request the server to redirect for us: mod_alias or mod_rewrite.
Alias /image /www/html/image
Redirect /service http://foo2.akfpartners.com/service

6. Use Caching Aggressively
Rule 20—Leverage Content Delivery Networks

Rule 21—Use Expires Headers
proxy caches don't inspect the HTML, they do not abide by these tags at all.

Keep-alives, or HTTP persistent connections, allow for the reuse of TCP connections for multiple HTTP requests.

Rule 22—Cache Ajax Calls
Rule 23—Leverage Page Caches
A page cache(reverse proxy server) is a caching server you install in front of your Web servers to offload requests for both static and dynamic objects from those servers.

ETags are unique identifiers issued by the server for an object at the time of first request by a browser. If the resource on the server side is changed, a new ETag is assigned to it. Assuming appropriate support by the browser (client), the object and its ETag are cached by the browser and subsequent If-None-Match requests by the browser to the Web server will include the tag. If the tag matches, the server may respond with an HTTP 304 Not Modified response. If the tag is inconsistent with that on the server, the server will issue the updated object and its associated ETag.

Rule 24—Utilize Application Caches
Rule 25—Make Use of Object Caches
Rule 26—Put Object Caches on Their Own "Tier"

Chapter 7. Learn from Your Mistakes
Rule 27—Learn Aggressively
Doing something without measuring the results or having an incident without learning from it are wasted opportunities that your competitors are taking advantage of.

Rule 28—Don't Rely on QA to Find Mistakes
Rule 29—Failing to Design for Rollback Is Designing for Failure ==> to-do
Rule 30—Discuss and Learn from Failures
http://www.cnblogs.com/xing901022/p/4425124.html

This book in many ways around the high scalability made 50 recommendations, a highly scalable site with the development of business, the increase in users, a free extended architecture, making it easy to cope with the rapid development of the site. Let's look at the specific content of the book:

YouTube
Hiredinte Desig
Question
The mos JobHu
Timer Ca
Pavel's B Sets C
N00tc0d3

► May (2)
► Februar
► January

► 2014 (115)
► 2013 (2)
► 2011 (8)

Popular Posts

Design a ch Runhe Tian

[CC150v5] 8 Shuatiblog.

thought-wor Elevator in a

Avro vs Prot

How to desi - GeeksforG

Design Hit (requests in Stack Overfl

Solr vs. Elas

The Uber sc

Design an i

Uber Desig

back to the top

Simplification of equation

1 Do not over-design

Excessive design corresponds to the system increases the complexity and cost of maintenance. These over-design, in normal use, but not much action. Designers often think themselves very important function or icing on the cake, the actual usefulness.

Taking into account the design of scalable 2

Designed at design principle to follow: when considering the design capacity of 20 times, three times the capacity to consider when implemented, considering the capacity of 1.5 deployment. When one side of the expansion of the project, the difficulties caused by the temporary extension.

3 A simple program again Jane

Should follow the Pareto principle, 20% of the design work done by 80%, so 80% of the time, should be placed in the 20% of the design.

In fact, a major function of the product are concentrated in several points, put the design several points, the other are some additional features only. So this core business must ensure that sufficient simple to use.

4 Reduce DNS lookups

Each different file domain, need to query DNS load time. For example cnblogs.com and i.cnblogs.com belong to different domains. Then query the DNS and they will query twice. When traffic is large, it will cause a certain impact.

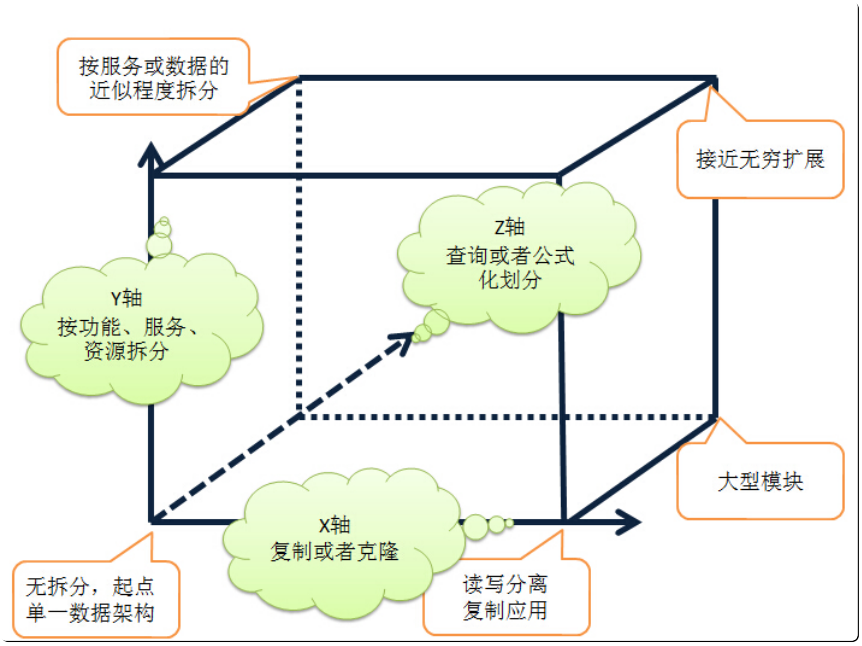
5 to minimize the object

Since the object when the browser access, you need to load. So it can consider a request to reduce the number of files (the number of concurrent browser to load the relevant number), the possible merger of some objects. For example, the icon class file that can be merged into a larger picture. Reasonable number of files, the browser will accelerate access to loading.

6 using the same brands of network equipment

Due to a http request, possibly through a lot of physical devices. Such as load balancers, switches, routers. So try to use the same brand of device, it will avoid some unexpected circumstances.

Distribution of work



7 X-axis, horizontal copy

Such things simple extension service, by cloning or replication to achieve, such as your applications on multiple servers and services. Common such as clustering, load balancing and so on, read and write database separation.

8 Y-axis, split something different

Large systems, split the different functions, such as registration, purchase, search, cloud disk. and many more

9 Z-axis, similar to the split of different things

For example, according to the user level, or the user's location, and so split.

Scale Design

10 is designed lateral expansion program

Extensions include horizontal, vertical. Transverse is by copying cloning applications, the use of minicomputer cluster expansion. Longitudinal is to increase server hardware and network infrastructure.

Through a lot of cases can be found, a simple upgrade hardware scale up, just a little to solve real pressure. Through horizontal expansion of the cluster, but can stretch the realization of freedom.

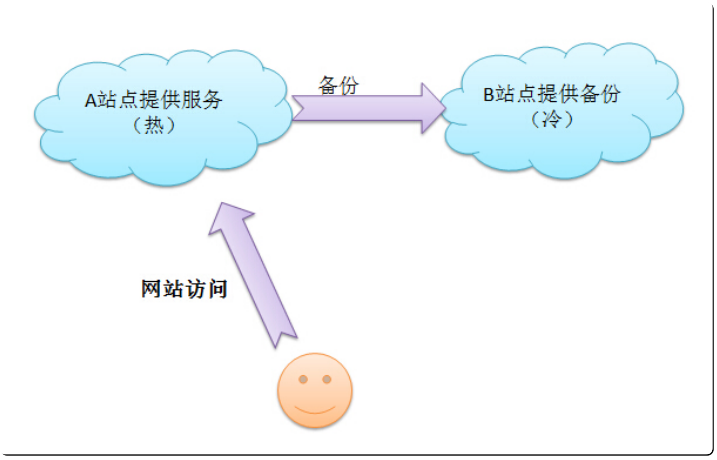
11 use of economic systems

Similar to the above principles, the use of high-priced server, does not guarantee good performance in the future. You should use common minicomputer cluster expansion.

12 scale data center

There are a lot of data center design, such as

Hot and cold station configuration: thermal station of service, when the collapse of the thermal station, using cold station to continue to serve.



Recommended sites using multiple real-time, lower cost, dynamic invocation. The disadvantage is the increased difficulty of operation and maintenance.

13 use of cloud technology in design

Cloud computing is virtualization a little, it can peak business, the elastic expansion devices. And daily treatment with the return of the extension.

The disadvantage is to improve the coupling applied to the virtual environment. Later refers to the use of physical devices, isolation operations in a virtualized cloud computing, service isolation might cause some interference to troubleshoot error

Use the right tools

14 rational use of database

There are many versions of the database, such as the traditional relational databases Oracle, MySQl, there is a relatively new non-relational database NoSql, such as MongoDB, and memory database FastDB, as well as specifically for SSD solid state drive Aerospike like.

But by the time of the selection, or to an individual business needs to be. See your database requirements of speed, or security and so on.

15 firewall, firewall everywhere

Firewall can intercept some invalid access filter. Usually put some CSS, static files, images, JS and so do not use a firewall, and critical business information when it comes to personal use. Rational design of the firewall will have some impact on the performance of the site.

16 actively using the log file

Using a variety of logs and tools, real-time monitoring of business. Just monitoring server memory CPU, but also on data services should be monitored. For example splunk (providing log

collection, storage, search, graphical display).

Do not do repetitive work

17 Do not just check the work done immediately

For example, just as a data write, do not read immediately. While some customers need to ensure the integrity of data can not be lost. However, you can log and other records, finished check this practice is not recommended.

18 Stop Redirection

Redirect will consume a certain delay, computing resources. Should be avoided

19 relax timing constraints

Most relational database stress ACID properties, it causes some problems when expanding. Therefore appropriate to relax the timing constraints in some business, you can improve your site's performance.

For example, a stand of booking, after subscription, will wait for the hotel's review. Such as a treasure, at the time of withdrawal, to confirm the scope of time. This is to expand the timing constraints, and to improve site performance and transaction security.

Active use of cache

20 use CDN

CDN customers can save data and content. Probably the process, when the user during the site visit, go to the CDN servers, CDN perform DNS queries, the user requests allocated to different servers. There are many CDN service providers to provide such services.

21 Using Expired head

For different types of objects, the use of expired head, reducing object request. Common HTTP corresponding property is: public no-cahe max-age, etc.

22 Caching Ajax call

The right to modify the Http header Last-Modified Cache-Control Expires other attributes.

23 use page caching

Cached response request before winter to reduce web server load.

24 use the Application Cache

For example, for some special users, their requests cached data.

25 use the object cache

Suitable for repeated use query data objects. For example, a shopping site, selling products cached data.

26 object cache on its own layer

Use a separate buffer layer, easy to expand and maintain.

Learn from our mistakes

27 active learning

The company has a learning atmosphere, will be derived from better products. Learning content on one hand, the customer's business knowledge, on the one hand from the field of technology and operation and maintenance.

Do not rely on QA found 28 errors

Employment testing or quality assurance personnel, the biggest purpose is to detect the correctness of the product. It can reduce costs, increase development speed of the developer because the developer does not need to always pay attention to the correctness of the code, it can be handed over to QA testing.

But QA responsible only found the problem, how to avoid the title still have to rely on developers.

29 is designed to be no rollback failed design

Here rollback, referring to the product release back. If you run some versions of BUG, you may need to be delivered before version can run at this time there is no fallback, you can not deliver the

product.

It is recommended the continuous integration of learning content.

30 to discuss and learn from failure

Should not fail twice on the same issue, summarized each failure is more indispensable.

Database Principles

ACID properties of a relational database:

Atomic: A transaction either all executed or not executed,

Consistency: the beginning and end of the transaction, all of the data state to be consistent,

Isolation: the performance of the transaction is the only transaction on the database operation,

Persistence: the transaction is completed, the operation can not be changed.

Note that the relationship between the cost of high 31

At the design stage should be a well-structured table design, and so the development started, some increase in the column, it may take a very high price.

32 Use the correct database lock

There are many concepts of database locks, such as lock implicit explicit locks, row locks, page locks, range locks, table locks, database lock, and so on.

Irrational use of locks, will affect the throughput of the site.

33 Do not use multi-phase commit

For example, two-phase commit: the first vote in the submission. This time reduces scalability, because before it is submitted to complete the transaction, can not be used for other operations.

34 不要使用select for update

Because the FOR UPDATE clause can cause rows to lock, reduce the speed of transaction processing.

35 Do not select all of the data

比如select * from xxx;

This practice is not the first open and extended data, such data would have been four, business processing code written directly die. When an increase in the data, it will result in error; the other is'll check out the unnecessary data.

或者inset into xxx values(xxxx);

This is when the column information does not match, it will also be wrong.

Fault-tolerant design and failure control

36 uses for fault isolation "lanes"

Service and data division there are many, such as container, clusters, pools, fragmentation, lane. Lane means that each business has its own domain, not across the lane called.

37 Do not trust a single point of failure

There are many systems designed as single-point mode, when the entire system is only using this module, when there is a single point of failure, the whole system will collapse.

Avoid system series 38

For example, a system has many components, each component security of 99.9%, when the series three components, the availability of the entire system becomes 99.7%.

39 to ensure the ability to enable / disable function

For some shared libraries, third party services should be provided on or off the function.

Avoid or distribution status

40 efforts to achieve a stateless

Realization of state will limit scalability, increased costs

41 as much as possible to maintain a session on the browser side

While reducing pressure on the server, on the other hand any request can be sent to any server.

42 use of distributed cache holds state

Use a separate cache layer, conducive to expansion. There are many distributed caching scheme, such as memcached.

Asynchronous communication and message bus

43 Whenever possible, use asynchronous communication

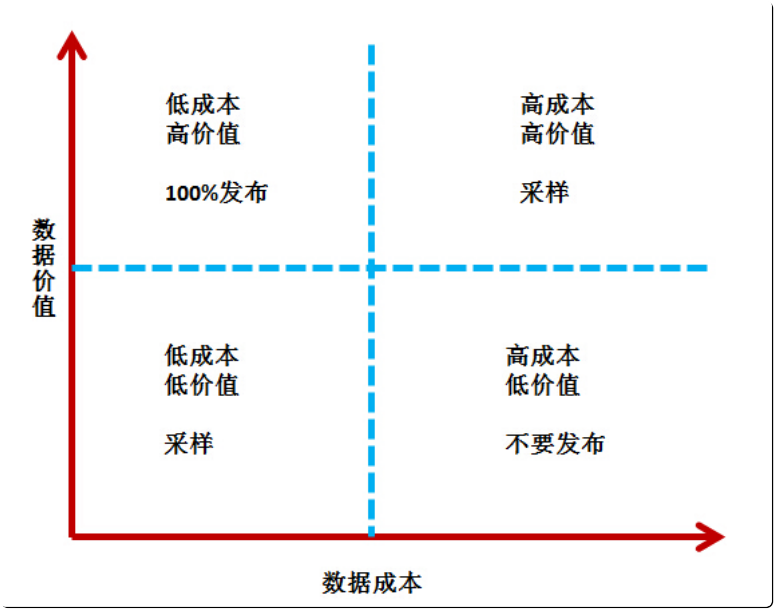
Asynchronous communication, to ensure the independence of each service and between the layers, so it is more easy early in system scalability and reduce the degree of coupling.

44 can be extended to ensure that the message bus

Maximize the use of the Y-axis or Z-axis expansion, ie business needs and extensions. Because simply duplicate, or clone, but will increase the number of listeners each message subscribers. By Activity isolation, can be separated business pressures.

45 Avoid overcrowding message bus

Measure the cost value and message.



Other principles

46 caution extended third-party solutions

If companies have problems, then find third parties to meet their immediate needs. But it is not a permanent solution, because the solution provider have many clients, your crisis is not their crisis, it is not possible at the crucial moment, due diligence. Therefore, enterprises should still have some control of power (the word is really tall!).

47 Clear, archiving and storage at a reasonable cost

There are some unnecessary data, you should regularly delete. Some slight value data periodically archive directly deleted. Some valuable data should be backed up as well as quick access.

48 delete transaction processing business intelligence

Should be separated from production systems and business systems, improve product scalability.

Avoid business expansion is limited by the system architecture.

Applications designed to monitor 49

Global monitoring policy should be designed to ensure that the answer

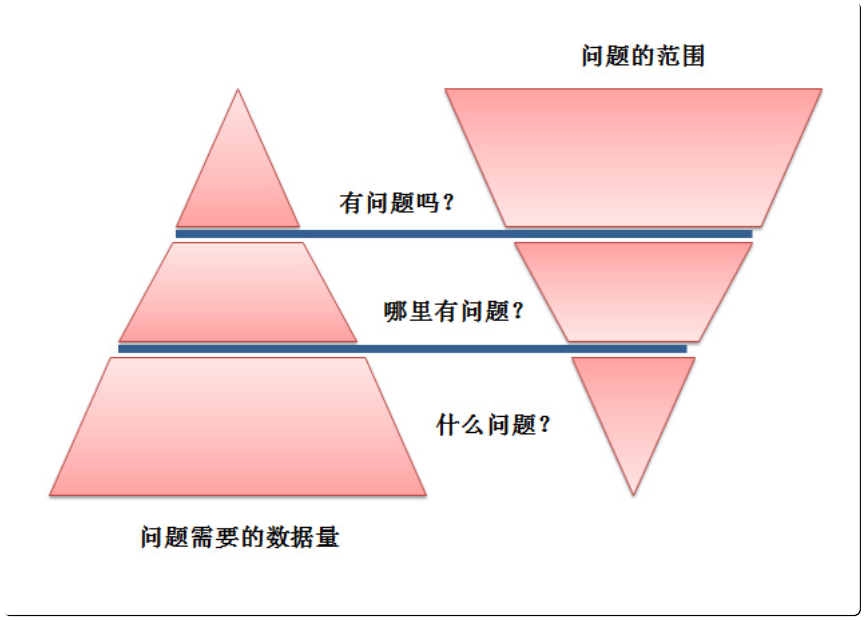
"A problem has occurred yet?"

"Where the problem occurred?"

"What happened?"

"The problem will happen?"

"You can automatically repair it?"



50 to be competent

Should relate to the most outstanding architecture in each design, we can not rely entirely on third-party solutions.

A simple good architecture, are small but excellent, if simply rely on open source solutions architecture that solves the problem, but it will lead to a bloated applications.

Read full book from Scalability Rules: 50 Principles for Scaling Web Sites

You might also like

- task dispatching system
- Brag Your Way to Job Interview Success
- Face questions a traffic light management system | Flyne
- Building Maintainable Software
-
- Application Services
- Cracking On-Site Code Interview
- Websocket
- Java ArrayDeque
- Linux Shell Scripting Misc 2

Posted by Jeffery yuan at 9:28 PM

Labels: Book Notes, Interview - Review, Scalability, System Design

No comments:

Post a Comment

Enter your comment...

Comment as: tejas49 (Google) Sign out

Publish Preview Notify me

Links to this post

Create a Link

Newer Post Slider(Newer 20) Home Random Post Slider(Random 20) Slider(Older 20) Older Post

Subscribe to: Post Comments (Atom)

Labels

System Design (156) Java (93) Interview-System Design (65) Book Notes (57) Interview (53) to-do (40) Linux (35) Interview-Java (32) Design Patterns (29) Product Architecture (27) Miscs (25) Big Data (24) Database (24) Cracking Code Interview (22) MultiThread (22) Interview - Review (21) Java - Code (21) Interview Q&A (20) Distributed (19) System Design - Practice (19) Algorithm (13) How to (13) Operating System (13) Brain Teaser (12) Architecture Principles (11) Company - LinkedIn (11) Google (11) Security (11) Code Quality (10) Concurrency (10) Testing (10) Tools (10) Redis (9) Architecture Model (8) Cache (8) Company - Uber (8) Java67 (8) Linux - Shell (8) OO Design principles (8) SOLID (8) Scalability (8) Spark (8) Interview - MultiThread (7) Java Basics (7) Resource (7) Web Dev (7) Amazon (6) Better Programmer (6) C++ (6) File System (6) Git (6) Highscalability (6) How to Ace Interview (6) Network (6) NoSQL (6) Search (6) Solr (6) Spring (6) Design (5) How to Better (5) How to Interview (5) Interview-Operating System (5) JDK Source Code (5) JavaScript (5) Math (5) Must Known (5) Soft Skills (5) Big Fata (4) C (4) Code Review (4) Company - Twiiter (4) Company Product Architecture (4) Design Principles (4) Facebook (4) Hardware (4) Hash (4) Interview Corner (4) JDK8 (4) JVM (4) Leetcode (4) Optimization (4) Product + Framework (4) Shopping System (4) Source Code (4) Trouble Shooting (4) node.js (4) Back-of-Envelope (3) Cassandra (3) Coding (3) Company - Pinterest (3) Consistent Hash (3) GOF (3) GeeksforGeeks (3) Generics (3) Google Interview (3) Growth (3) Guava (3) Interview-Big Data (3) Interview-Linux (3) Interview-Network (3) Java EE Patterns (3) Javarevisited (3) Kafka (3) Map Reduce (3) OOD Design (3) Performance (3) Puzzles (3) Python (3) Resource-System Desgin (3) Restful (3) UML (3) Web Service (3) geeksquiz (3) AI (2) API Design (2) AngularJS (2) Bugs (2) CareerCup (2) Data Structure Design (2) Data structures (2) Database-Shard (2) Debugging (2) Elasticsearch (2) Garbage Collection (2) GeoHash (2) Hadoop (2) Html (2) Interview-Database (2) Interview-Miscs (2) Interview-Web (2) JDK (2) Logging (2) Machine Learning (2) Math - Probabilities (2) POI (2) Papers (2) Programming (2) Project Practice (2) Random (2) Scala (2) Software Desgin (2) System Design - Feed (2) Thread Synchronization (2) Video (2) reddit (2) Ads (1) Advanced data structures (1) Android (1) Approximate Algorithms (1) Base X (1) Bash (1) Be Architect (1) Behavior Question (1) C# (1) CSS (1) Chrome (1) Client-Side (1) Cloud (1) Coding Interview (1) CodingHorror (1) Company - Facebook (1) Company - Netflix (1) Company - Yelp (1) Counter (1) Crawler (1) Cross Data Center (1) DSL (1) Dead Lock (1) Difficult Puzzles (1) Docker (1) Eclipse (1) Facebook Interview (1) Function Design (1) Functional (1) Game Design (1) Go (1) GoLang (1) ID Generation (1) IO (1) Important (1) Internals (1) Interview - Dropbox (1) Interview - Project Experience (1) Interview - Soft Skills (1) Interview Tips (1) Interview-Algorithm Desgin (1) Interview-Brain Teaser (1) Interview-How (1) Interview-Mics (1) Interview-Process (1) Jeff Dean (1) Joda (1) LinkedIn (1) Mac (1) Micro-Services (1) Mini System (1) MySQL (1) Nigix (1) NonBlock (1) Productivity (1) Program Output (1) Programcreek (1) RPC (1) Raft (1) RateLimiter (1) Reactive (1) Reading (1) Reading Code (1) Resource-Java (1) Resource-System Design (1) Resume (1) SQL (1) Sampling (1) Shuffle (1) Slide Window (1) Spotify (1) Stability (1) Storm (1) Summary (1) System Design - TODO (1) Tic Tac Toe (1) Time Management (1) Web Tools (1) ZooKeeper (1) algolist (1) corejavainterviewquestions (1) martin fowler (1) mitbbs (1)

Popular Posts

Design a chess game using OO principles | Runhe Tian Coding Practice
http://k2code.blogspot.com/2014/03/design-chess-game-using-oo-principles.html http://swcodes.blogspot.in/2012/09/chess-game-design.html ...

[CC150v5] 8.4 Design a Parking Lot - Shuatiblog.com
[CC150v5] 8.4 Design a Parking Lot - Shuatiblog.com A parking lot has multiple Levels. A Level has multiple Parking Spot. A Spot can pa...

thought-works: Object Oriented design for Elevator in a multi-storied apartment
thought-works: Object Oriented design for Elevator in a multi-storied apartment A typical lift has buttons(Elevator buttons) inside the ca...

Avro vs Protocol Buffers vs Thrift

https://martin.kleppmann.com/2012/12/05/schema-evolution-in-avro-protocol-buffers-thrift.html Thrift, Protobuf and Avro all support schem...

How to design a tiny URL or URL shortener? - GeeksforGeeks

How to design a tiny URL or URL shortener? - GeeksforGeeks How to design a system that takes big URLs like "http://www.geeksforgeeks....

Design Hit Counter - how to count number of requests in last second, minute and hour - Stack Overflow

algorithm - how to count number of requests in last second, minute and hour - Stack Overflow There is a hypothetical web server which supp...

Solr vs. Elasticsearch

-- I already uses Solr for several years, here just try to know more about Elasticsearch and the difference and Elasticsearch's advanta...

The Uber software architecture

http://highscalability.com/blog/2015/9/14/how-uber-scales-their-real-time-market-platform.html Video http://www.infoq.com/br/presentation...

Design an in-memory file system

http://www.careercup.com/question?id=13618661 1> Use Structure with n children, on parent pointer , one data field and one name field. ...

Uber Design Excel

http://codeinterviews.com/Uber-Design-Excel/ How would you design an Excel sheet's Data structure. You need to perform operations like ...