Week 1, Backend for High Load

Azamat Serek, PhD, Assist.Prof.

High load systems

High load systems became a trend back in 2012. But there's a problem with them - we still have no clear definition of the term. Where do high loads start? Are 10 requests per second already high load or not yet? And what about 100 requests or 1000? You might be surprised, but the numbers are not the point here at all.

What is high load

First, it is important to understand one simple axiom: high loads are a relative concept. They cannot be measured by the number of requests that arrive at the server, or by the speed of the website. After all, there is no “average” number of requests, as well as an “average” site. One web resource will be able to process a thousand requests per second without any difficulties, while the other will crash on the hundredth connection. So the point here is not at all in quantitative indicators.

High load definition

High load means the resistance of an Internet project to high loads. But wait, there’s more. This is not some universal piece of code that we copy-paste and after which everything flies. This is setting up the site architecture: working with databases, a server, using modern technologies and programming languages.

If we draw an analogy with an ordinary clothing store, then instead of servers, programming languages and IT stuff like that, there is a simple and understandable consultant, cash register and goods. On a typical day, a consultant approaches each client, helps to choose the size, advises accessories, then escorts to the checkout and calculates the buyer. Two or three consultants can do it quite well.

High load

We have compiled the most popular high load definitions from IT professionals and users who are knowledgeable in this topic:

- Highload is when the IT-system ceases to cope with the current load.
- Highload is when traditional approaches to the work of the IT infrastructure are already not enough.
- Highload is when one server is not enough for customer service.
- Highload is when techniques can not cope with the increased loads.
- Highload is when problems that arise cannot be solved by available means.
- Highload is when the infrastructure needs to be urgently scaled.

Biggest high load challenges.

High load infrastructure processes large volumes of data and thereby generates great value for a business. Due to this, failures and other quality problems result in the extra cost for companies. Thus, according to the Gartner article, the loss of large online services reaches $ 300,000 per hour in case of downtime.

A key source of problems in high load infrastructure is the volume of data, complexity and rate of change. Therefore, it is important that the overall high load architecture of a large application should be designed both in terms of the software components and the hardware on which they operate. Moreover, when designing a custom high load system, it is necessary to understand clearly what delivery times are set, what legal restrictions are, how much experience the specialists involved in the design have. Besides, we should be fully aware of the associated risks and their acceptability for the company.

Questions

1. how to ensure the correctness and completeness of the data, even in the event of failures;
2. how to maintain the high app performance for system users;
3. how to scale up in case of load growth; etc.

Principles to take into account

- Availability.

Uptime is directly correlated with the reputation and performance of many companies.

- Performance.

The speed of a web resource affects user satisfaction with the service, as well as ranking in search results (which is reflected in traffic).

- Reliability.

The request should always return the same data to users so that users are sure that if some data are recorded/entered into the system, during subsequent extraction, you can count on their invariability and safety.

- Scalability.

We can talk about various parameters of the system: how much additional traffic it can handle, how easy it is to increase the storage capacity, how many transactions can be processed in excess of the current capabilities.

- Manageability.

The development of a custom high load system that is easy to operate is extremely important at later stages of project development (easy diagnosis and understanding of the essence of problems when they arise, easy updating or modifications).

- Cost.

Includes hardware and software costs. It is important to take into account other aspects necessary for the deployment and maintenance of the system: the amount of time spent by the developers on assembling the system, the amount of effort required to start the system, training, staff training, etc.

Crashes

The main problems in the design of custom high load systems arise in the following segments: data volumes, data dissemination, data correction, use of open-source software, search, processing & analysis of data and information modeling.

Important: You need to pay more attention to the reliability of high load systems. Reliability means the ability of the system to continue to operate normally even in case of a problem. The problems that arise are called failures, and systems built with them are called failures, too.

Talking about the reliability of high load systems, it is necessary to mention the fault management documentation. Well-written crash management documentation should include a simple step-by-step guide to recovering your system from almost any possible crash.

Redundancy

If we talk about global redundancy, then all the rules of interaction between servers also apply to data centers - we must have a margin of safety (capacity, computing power, etc.) in order to continue to work with the loss of one data center without significant damage to quality services provided.

To ensure system reliability, it is recommended to use the following approaches:

- Separate those parts of the system that affect the performance of the system from the parts most prone to human error.
- Implement all forms of testing, be it unit testing, complex system testing or manual tests.
- Provide tools to recover the system in the event of a failure as soon as possible to minimize the impact.
- Implement a system of metrics, monitoring and logging as tools for diagnosing errors and causes of failures.

5 benefits of a custom high load system.

01

It is a system with a huge audience

If we talk about web applications, then these are thousands, and sometimes hundreds of thousands of people. Of course, a specific figure cannot be called. But it is clear that an online store that processes 10 requests per day cannot be called a high load, but Facebook, Amazon, Flickr, MySpace or Youtube certainly can.

02

This is a distributed system

If the application has to process huge amounts of data, which is also constantly growing, one server is not enough. The largest high load app solutions like Google or Facebook work on hundreds of servers.

But a huge number of machines are caused not only by high loads. More precisely, not only with a large number of requests that have to be processed non-stop. At the same pace, the servers quickly fail, so the more there are, the higher is the likelihood that the system will quickly recover after a failure.

This is a system with positive dynamics

If an online-offer is valuable for users, its audience is growing. Therefore, the high load is not just a system with a large number of users, but a system that intensively builds an audience.

04

This is an interactive system

If a person enters a search query on Google, uploads a video to YouTube or makes a purchase on eBay, they expect to receive the result immediately. If the system takes a long time to respond, most likely they will start searching somewhere else. Therefore, instant response is a distinctive and very important feature of a high load system.

05

It is a high-resource system

This item is directly related to the previous one. To give an instant response, the system goes through a lot of resources - CPU, RAM, disk space, etc. And for this, it is necessary that the resources: a) be free and b) be in sufficient quantities (preferably even with a margin).

Here we come to the paradox of high load systems: the faster they grow, the more stringent it is to control resources. When an application grows its audience, the number of requests naturally grows. And the number of resources that need to be spent to maintain interactivity also grows with them.

That is, the high load is a system that needs to be constantly scaled. Setting it up to work in this way is quite difficult, but from a business point of view it is worth it.

Why consider high load system development?

In general, it is necessary to determine two major things. The first one is how large the audience that the project can face is expected to be. Secondly, the project will have to work with a structured data set, so the second important thing is to understand how large and complex this structured data set is going to be.

But it's better to start from the types of projects, for which using the high load approach is highly recommended:

Information resources: in the overwhelming majority of cases, the key point here is that there is no need to plan the high load architecture in advance, a very small amount of information resources outgrows the capabilities of one server with a well-tuned content management system. And even if this happens, adding new servers will be painless due to the wide possibilities for caching content and no need for complex solutions at the DBMS level.

Online stores: everything here rests on the breadth of the assortment - if the assortment theoretically cannot grow to hundreds of thousands of items, then no special decisions will be required. Otherwise, you need to think about efficiently caching frequently visited products and categories, as well as minimizing "heavy" queries to the database.

Social networks: the main criterion is the theoretically possible number of participants and the connections between them. If we are talking about a small regional or narrow-topic social network, then you can easily put everything in a relational database, but for something bigger you will need a scalable solution.

Technological projects: projects based on technical know-how directly depend on it and, as a rule, already at the stage of designing the "highlight" of the project, it becomes clear which architecture is more suitable for it and how to develop it in the future. High load application architecture.

Architecture is the foundation of an application. And as in construction, the quality of the house depends on the strength of the foundation, the success and viability of the system in the development also relies on the same.

In our decisions to use or not to use high load systems, we focus on what a particular business needs. But there is also planning - something that the business does not see and from which it does not directly benefit. When it comes to high load applications, it is not enough to develop a competent architecture, you also need to build a monitoring system nearby that will monitor the viability of components, the speed of reaction, generate and display data to monitor the health of the high load system.

What will happen if the monitoring system is not implemented in a high load system?

A high load application can behave unpredictably and stop working in the most unexpected place. Moreover, the system will do this at the time of peak load: at the time when the business earns the most. Therefore, saving on the monitoring system is illusory. If you abandon it, you can end up losing a lot of money.

References

https://geniusee.com/single-blog/introduction-to-high-load-what-is-it