BUILDING SCALABLE E-COMMERCE SYSTEM

E-COMMERCE & OMNI-CHANNEL

AUTHOR:

Sandeep has more than 17 years' experience and has been working with HCL Technologies for more than 7 years. He has executed and delivered a number of complex technical deliveries and is a part of the retail-multichannel commerce stream and provides consultation for performance, scalability and resiliency issues across deliveries.

TABLE OF CONTENTS

Able to scale with demand and time is the key to the success of an e-commerce application

1. INTRODUCTION

Over the past two decades, e-commerce applications have evolved from a single application supporting single channel to multiple ecosystems supporting multiple business channels. Now a day's shopping begins with a virtual store browsing and concludes also at the webstore.

A study* showed that 48% of all retail sales are either online purchases or web-influenced purchases.

With increase in touch points, the traffic these systems needs to support has grown exponentially. The e-commerce sites downtime can lead to negative coverage by media. Hence loss of revenue, or loss of shareholder equity. Therefore, it is imperative that scalability be a key design principle for all e-commerce applications.

Below is a depiction of all the touchpoints many organizations now have with their customers.

2. WHAT IS SCALABILITY

There are number of definitions on web for scalability. Let me define it in a simplified way.

Scalability is the ability of an underline system to increase its throughput (e.g the concurrent number of users to be supported) by adding some extra resources (e.g. hardware). The intent is to increase the throughput in direct proportion of resources added to the underline system. There are two ways a business can achieve scalability.

Note: It is important not to mix the two terms Throughput and Scalability as the two are different but related.

Vertical Scalability: In vertical scalability we increase the physical resources of the system like CPU or Memory. If the marginal input (e.g. hardware) equals the marginal output (e.g. throughput in terms of page views/second), the resource can be considered a perfectly scalable else different option/configuration needs to be looked upon.

Scalability should not be mistaken as Throughput as the two terms are totally different and are used in the context of a performing e-commerce application

Vertical Scaling is addition of hardware resources to an existing resources. Usually in this the overall hardware topology doesn't change for overall application landscape

Horizontal Scaling is achieved by adding more resources/servers in the hardware topology

While vertical scaling can increase the overall throughput, limited redundancy can sometimes be a negative consequence. If a system rely solely on vertical scalability to deliver the required system throughput the failure of the single node would lead to the stability and availability of the entire system.

Horizontal Scalability: Ability to gain throughput by adding more resources (usually physical servers) as opposed to increasing the capabilities of an existing resource. An example of horizontal scaling would be adding an extra server for database as opposed to increasing the RAM/Memory of the existing database server.

To achieve true horizontal scaling, a system can have no shared resoruces. This is known as a "shared nothing" architecture. An individual component of the architecture is as scalable as the entire architecture. Further work needs to be done to remove potential bottlenecks, both by changing how the application interacts with shared resources as well as working to make those shared resoruces horizontally scalable.

Throughput

Throughput refers to the amount of work or capacity that a system or resource can perform or handle. The term should not be confused with scalability as the two are totally different. A web machine that can serve five pages per second may be more scalable than a machine that serves 50 pages per second. If the marginal input (e.g. hardware) equals the marginal output (e.g. throughput), the resource as a whole is considered scalable.

Reduced throughput is usually the first sign of scalability problem and so it is important to track it over time for each resource.

Although there are no definite rules for ensuring the scalability of an e-commerce application but some common/best practices can be seen/evaluated/implemented for achieveing the said scalability in a system

Caching is both powerful and erroneous options. Proper analysis, justification needs to be in place for what and when needs to be cached.

Also needs to define the refresh strategy and monitoring of cached data for the overall system performance

3. SCALABILITY BEST PRACTICES

Although there is no defined set of rules/practices that can at the beginning ensure that the system will be scalable enough, but the following points should be considered while designing to achieve scalability to large extent at the starting:

1. **Decouple:** Shared resources are single point of failures. For example, consider a web service hosted on a single machine. In real time if synchronous calls are made to this web service, the system under is called the shared resources and its failure will question the availability of the entire system.

   While designing the e-commerce applications, consider which module needs to be synchronous and which can be deferred for later (asynchronous). For example, during the peak hours the system should only be taking up user's request and orders (synchronous).

   Secondly, during the designing phase one needs to define when, where and how the work will be performed. The processing that is more important and time-sensitive can be scheduled to run ahead of the other less important processing. For example, we can design the system to submit orders to Order Management System on each instance of production with separate dedicated cluster for purging orders.

Another example of decoupling is SMTP e-mail server. Usually they are low-end servers used only for sending mails. If all the order process is made synchronous then it is quite frequent when during the peak time the SMTP server may go for a toss because of heavy user order requests.

2. **Caching:** E-commerce sites are very rich in content. Each web page contains tons of images and scripts besides regular text. Further, to provide a wide variety of product offerings, the databases and other storages are huge. In such a scenario, bringing all the data from backend storages to consumer browser in a reliable and efficient manner is a mighty challenge. A combination of multilayered and distributed caching technologies can help address this challenge. The key take away is that much of the server data is static and much of the dynamic data is not chaning so frequently.

    a. First level of caching can be done in the browser itself. The cached data lives on consumer desktop till it becomes stale or expires.

    b. Next level of caching is done using Content Delivery Networks, Akamai CDN. Internet traffic, from the browser to website makes several hops across multiple ISP's routers. Each hop adds apporx 100 millisecond in response times. CDN reduces hops by partnering with ISPs and placing their data just next to ISP (reduced hops for data).

NoSQL databases are also good candidates for acting as a secondary-level cache.

    c. Next use reverse proxies, such as apache traffic server to serve dynamic page content that doesn't change real time. Reverse proxies sit right in front of webserver and provides fine grain control over what is cached and when. End effect is reduced load on webservers, which now have to build dynamic pages less frequently.

    d. Fourth level of caching, can be done in application processing themselves. Today's 64-bit processes offer terra bytes of virtual address space. This translates to serious big caches limited only by availability of RAM on the machine. Earlier 32-bit allows max of 1.5 GB for application use. It is highly recommended to not exceed the heap size as greater the heap size the more time taken by JVM during the full GC.

    e. Next could be use of NoSQL databases, now non-structured databases can be used as secondary-level caches or even authoritative data sources. These databases offer high performance and scalability as compared to relational databases. Each NoSQL DBs has its strength on parameters such as read/write ratio, persistence , search/indexing capabilities. Pick the one with your choice of combination.

    f. Next use replica of master databases, when database becomes large, traffic to the database can be split with all updates directed to the master and while all queries are directed to one or more slave replicas.

With the use of a proper APM tool in place it becomes easy to debug the right areas/components that hinders the overall scalability or performance of the system

3. Multilevel Monitoring

Good and efficient monitoring is very critical for a top performing e-commerce site. In general, it ensures high availability and fast response times. Furthermore, it allows one to see the impact of changes quickly. This in turn allows rapid deployment of new features. Third, it allows developers to troubleshoot production problems quickly. This is especially important because the really complex problems show up only in production. Finally, it allows capacity planning in advance.

For implementing a good monitoring system, monitor each components independently. Unless there is strict control of SLA's between components, overall SLA cannot be achieved. Within components, measure everything at all levels and collect as much data as one can. Modern-day application have multiple levels of abstraction. Collect detailed data from all levels: network, operating system, jvm, application server, database, etc. A combination of system monitoring tools such as nagios, application monitoring tool through bytecode instrumentation such as Intrascope, New Relic or Hyperic, external site monitoring tool such as site 24x7 and application and jvm internal statistics would make a powerful monitoring setup. Finally, build an integrated view of the entire platform with monitoring feeds from individual components.

With now a days availability of Cloud platform by many Service Provider is acting as a good option for hosting the e-commerce application.

The main advantage of hosting on cloud is quick ramp up of hardware based on the projected demand and quick ramp down based on decreased demand/off season. This helps him to achieve high scalability with minimal budgets variations and get rid of hardware procurements and AMCs.

4. Host it on the Cloud

Building an e-commerce platform on cloud have some great advantages. First, it allows the team to focus on future development instead of worrying about hardware/infrastructure issues. Second, as business scales, scaling the technology platform on cloud is the easiest. Bringing up new server or adding more CPU or RAM can happen in few minutes/hours. This helps the business team to ramp-up/ramp-down the h/w resources based on the peak/off-peak seasons. Finally, when it comes to batch processing capability it is most cost effective.

5. Use Java and Open Source

When it comes to deciding which programming language to use, three are most common: php, ruby and java. In general, in a distributed architecture all 3 could co-exist. However, each have their distinct advantages. PHP and Ruby will get up and running much faster than java. Code maintainability is bet in ruby. However, I will choose java because when it comes to scalability, development and diagnostic tools and good/sufficient engineers who know how to tackle complex performance issues, java is far more mature platform.

With respect to open source, I think entire e-commerce platform can successfully be built in it. Linux, MySQL, Apache., Spring, Hibernate and many NoSQL DBs are widely used. Further, the active community forums also make up professional support.

6. Scaling e-commerce databases

Databases can be scaled horizontally and vertically. Generally customers do both, achieveing both high overall scalability and through horizontal scaling, high availability.

Database is the backbone of any e-commerce application and equal importance needs to be given for making it scalable.

A proper balance needs to be maintained for fetching and storing master and transactional data.

a. Database Offload

Use database replication strategies for example if Oracle is used as the database for an e-commerce application then use Oracle Data Guard or Oracle GoldenGate, one can push read-only data from one's highly available, writable database to a farm of separate non-clustered, known as slaves. Read-only data consists of data such as categories, products, SKUs and media. Data sources can be configured to pull all of this read-only data from slave database, with different groups of application server instances reading from different slaves. E-commerce reporting application can run from these slaves. Not having to query from main clustered database for many of the repetitive read-only queries can significantly reduce the load. Since slaves are read-only copies they do not need to be highly available or use any of the costly technologies to achieve high availability and scalability. These slaves can fail without any data loss or impact to the end customer.

b. Load Balancing

When establishing connections to database from application server, one can choose to enable or disable load balancing. Enabling load balancing allows databases to load balance traffic between dataabase clustered nodes. Connections that are mostly read/write can significantly increase interconnect traffic. Excessive interconnect traffic can lead to database performance and throughput degradation and should therefore be avoided. For connections that are mostly read/written, disable load balancing but turn on failover. With HTTP sessions stickness, any write operations to a database by a given user will happen to the same instance of the database in a cluster.

Connections that are largely read-only, like to the catalog schemas, can use load balancing.

7. Partioning Data

In addition to splitting up data by type, data of the same type can also be split using partitioning. Good candidate tables may include e-mail registration tables, order line items tables, or JMS-related tables.

Partitioning a table allows its data and indexes to be subdivided into smaller pieces, which can improve SQL query performance, maintenance and management in certain circumstances. Queries that take minutes to execute can be shortened to seconds or even sub-seconds. Partitions can be stored in different tablespace and created/modified/deleted at runtime, with no impact on performance or stability. As dataset grows, one can rebalance partitions at runtime.

Another use of this technology would be partitioning order table by date range with the older partitions running on progressively less expensive and available hardware.

8. Operational Considerations

Problems in database like locking, blocking or deadlocking sessions are likely to cause serious problems for a large e-commerce application in production. The threads in application server that are executing those queries hold on to Java-level object locks, which are not released until the application receives a response from the database. To minimize this impact, set the transaction timeouts low in the application server

In addition to a transaction timeout with thread interruption on the application server, each database should implement a script to automatically kill transaction and individual DML statements that run for too long. The script should take into account the sessions application as different applications have varying transaction and DML execution length.

4. CONCLUSION

The effort for making the ecommerce application scalable should start from its inception and design, and should be followed holistically till the entire lifecycle of the product. Although there are no set of defined rules/practices which can ensure the required scalability of the application but to start with one can follow the above guidelines/practices as these will definitely add to the improvements in the scalability. There are other factors from hardware perspective that should be considered for scalability but that is not in the scope of this whitepaper.

Besides this, regular load testing, APM tool implementations, and analysis of load and projected/peak volume times are must have activities to stabilize the application from scalability perspective.

REFERENCES

1. http://www.ibm.com
2. http://www.oracle.com
3. http://www.theserverside.com
4. http://www.slideshare.net/AryaShree1/ecommerce-web-app-architecture-and-scalability
5. http://www.rackspace.com/knowledge_center/whitepaper/building-your-ecommerce-strategy
6. http://www.p2080.co.il/go/p2080h/files/3418332755.pdf
7. http://aws.amazon.com/architecture/
8. http://www-01.ibm.com/support/knowledgecenter/#!/SSZLC2_7.0.0/com.ibm.commerce.developer.doc/concepts/csdsearchperf.htm
9. http://www.theserverside.com/report/Maximize-system-scalability-with-Scalability-Rules

ABOUT HCL

About HCL Technologies

HCL Technologies is a leading global IT services company working with clients in the areas that impact and redefine the core of their businesses. Since its emergence on the global landscape, and after its IPO in 1999, HCL has focused on 'transformational outsourcing', underlined by innovation and value creation, offering an integrated portfolio of services including software-led IT solutions, remote infrastructure management, engineering and R&D services and business services. HCL leverages its extensive global offshore infrastructure and network of offices in 31 countries to provide holistic, multi-service delivery in key industry verticals including Financial Services, Manufacturing, Consumer Services, Public Services and Healthcare & Life sciences. HCL takes pride in its philosophy of 'Employees First, Customers Second' which empowers its 104,184 transformers to create real value for customers. HCL Technologies, along with its subsidiaries, had consolidated revenues of US$ 5.8 billion, for the Financial Year ended as on 31st March 2015 (on LTM basis). For more information, please visit **www.hcltech.com**

About HCL Enterprise

HCL is a $6.8 billion leading global technology and IT enterprise comprising two companies listed in India – HCL Technologies and HCL Infosystems. Founded in 1976, HCL is one of India's original IT garage start-ups. A pioneer of modern computing, HCL is a global transformational enterprise today. Its range of offerings includes product engineering, custom & package applications, BPO, IT infrastructure services, IT hardware, systems integration, and distribution of information and communications technology (ICT) products across a wide range of focused industry verticals. The HCL team consists of over 109,643 professionals of diverse nationalities, who operate from 31 countries including over 505 points of presence in India. HCL has partnerships with several leading global 1000 firms, including leading IT and technology firms. For more information, please visit **www.hcl.com**

www.hcltech.com

Hello there! I am an Ideapreneur. I believe that sustainable business outcomes are driven by relationships nurtured through values like trust, transparency and flexibility. I respect the contract, but believe in going beyond through collaboration, applied innovation and new generation partnership models that put your interest above everything else. Right now 105,000 Ideapreneurs are in a Relationship Beyond the Contract™ with 500 customers in 31 countries. **How can I help you?**

Relationship™ BEYOND THE CONTRACT

HCL