Pushing Performance Boundaries For Ebay Shopping Cart

Shilpa Gopalakrishnan
shigopalakrishna@ebay.com
Ebay Inc

Shylu Arjunan
sarjunan@ebay.com
Ebay Inc

Charuroopa Varadarajan
chvaradarajan@ebay.com
Ebay Inc

Achyut Kulkarni
ackulkarni@ebay.com
Ebay Inc

ABSTRACT

The eBay Shopping Cart poses unique challenges unlike its counterparts due to its multi-faceted functionalities. It contains various flavors of items that are unqiue in the eBay ecosystem such as *Auctions*, *Negotiated* items and *Committed* items besides the regular items. Also, unlike several other ecommerce websites, the Shopping Cart page on eBay shows the user complete pricing, shipping, discount and tax details. Hence, keeping the Cart updated in real-time with accurate pricing is a very complex problem.

This paper presents a STAR architecture model along with various optimization techniques which resulted in staggering performance improvement to eBay's Shopping Cart and thus enhancing the user experience. We also emphasize the importance of Elastic search coupled with Kibana and Grafana as performance evaluation and debuggability tools.

1. AUDIENCE

While we do not restrict this paper to any specific audience, we will assume basic Computer Science fundamentals. Any background in Database design, Near and real-time systems, Distibuted or Microservices architecture and/or Monitoring systems will also help.

2. INTRODUCTION

eBay has a unique business model which allows a user to bid on "Auction" items and "Commit" to buy items without having to pay immediately. Apart from showing the regular items, eBay's Shopping Cart also shows these Auction and Committed items. This user experience, while very beneficial for the user, becomes a disadvantage for the performance of shopping cart.

eBay's cart unlike most e-commerce sites, provides complete pricing details including discounts, shipping and tax. Users are also given the flexibility to modify shipping options, negotiate prices with sellers and pay for items from a single seller or the entire cart. This flexibility adds a heavy dependency on downstream services which further hampers the performance.

2.1 Challenges

- **Consistency - A sum of parts:** The user commits to buy items and bids on Auction items on non shopping cart pages on the eBay site. The cart needs to be updated instantly with these committed items and auction items that the user wins which require offline processing to maintain consistency.
- **Accuracy - Dependency hell:** In order to show complete price and shipping information on the cart page, several downstream services have to be invoked. Each service can have further fan outs into their own set of dependencies which is typical of a distributed system such as ours.
- **Size - Cart or Wish list?:** A lot of our Buyers use the Cart as a "wish list", using it to keep all the items they would be interested in and then come back to the Cart when they are eventually ready for their purchase. In that time, a lot of these items could have changed cost, quantity or simply become unavailable which makes consistency and accuracy a greater challenge.
- **Metrics - Shooting in the dark:** The first step to improve performance is to have the right tools to measure it. Shopping cart being a legacy system lacked the capability of measuring real time analytics.

The paper describes a scalable solution for the aforementioned challenges and also the results showing the significant performance improvements that were achieved.

3. METHODOLOGY

3.1 Internal Architecture - A STAR model

The older architecture of Ebay's Shopping Cart mimicked its visual representation which as depicted in **Figure 1** . is a hierarchical design with seller group at the highest level and Items at the lowest. Any operation on an item had to traverse through these levels with $\mathcal{O}(n^4)$ which was highly inefficient.

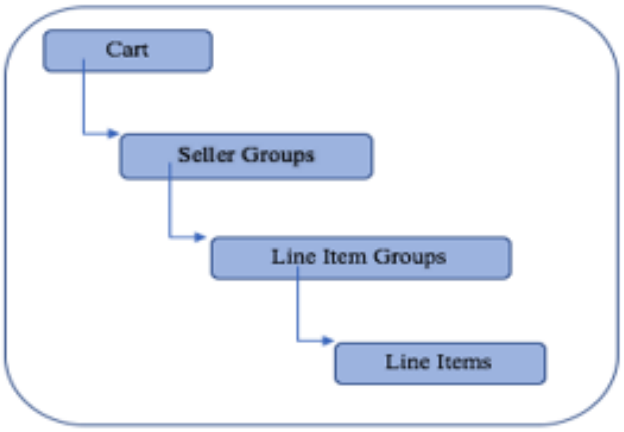

Figure 1: Old Shopping Cart Structure

Such time inefficient systems could be significantly improved with a trade-off between space and time which leads us to our newer design called STAR MODEL(**Figure 2**). This system breaks the hierarchical structure by keeping the items in the centre and every item related information at a one-hop distance achieving a time complexity of $\mathcal{O}(1)$. Although this increases the space complexity, the recent advancements in storage technology make this a fair trade.

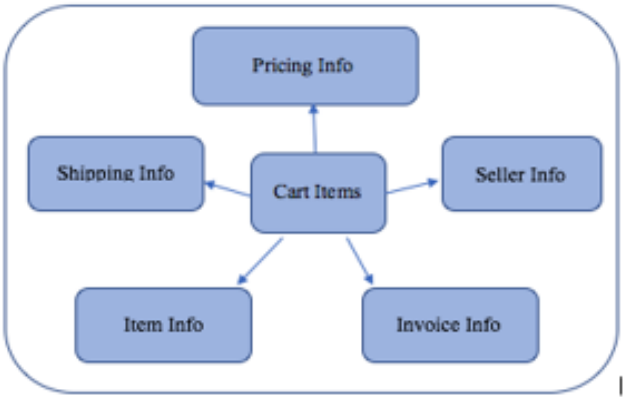

Figure 2: New Shopping Cart Structure

3.2 Optimization

The pricing and shipping information for shopping cart is obtained from a downstream pricing service. The Shopping cart response time is directly proportional to the response time of the Pricing Service. Performance metrics indicated that the Pricing Service response time increases linearly with increase in cart size as depicted in **Figure 3** . Also it constituted 65% of the total response time and hence was identified as the major bottleneck to the cart performance.

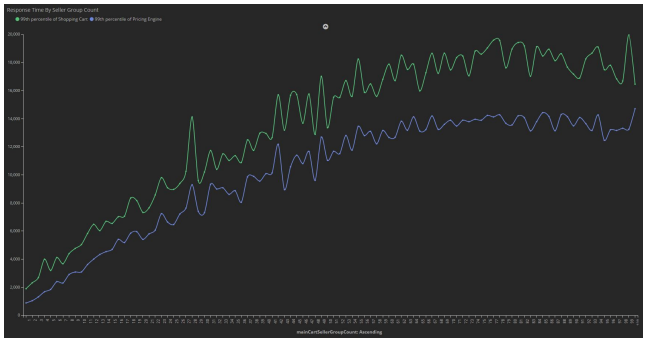

Figure 3: Shopping Cart Response time(Green) and Pricing Service response (Blue) vs Seller Group Count

To break the dependency of Shopping Cart's SLA on Pricing Service response time, we designed a mechanism to recalculate price information only for the dirty (modified) seller bucket as opposed to the earlier approach where we recalculated price for all seller buckets in the cart. Essentially this would mean that, irrespective of the cart size, the Pricing service response would be at par with that of a cart with only one seller bucket.

For instance, if a user adds or updates an item in their cart, Pricing will only be calculated for the seller bucket containing that item. The results of this optimization were remarkable. **Figure 4**. depicts the response time of Regular call vs Optimized call of Pricing service.

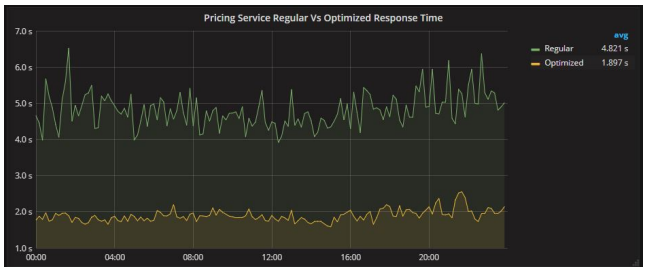

Figure 4: Regular call vs Optimized call of Pricing service

As a result of the above optimizations, we achieved a constant response time for Shopping cart service irrespective of the number of seller buckets in the cart. **Figure 5** and **Figure 6** depict the comparison between the Legacy and Optimized Cart response times. In these graphs, we plot the 99th percentile of Shopping cart response time against the seller bucket count in cart.

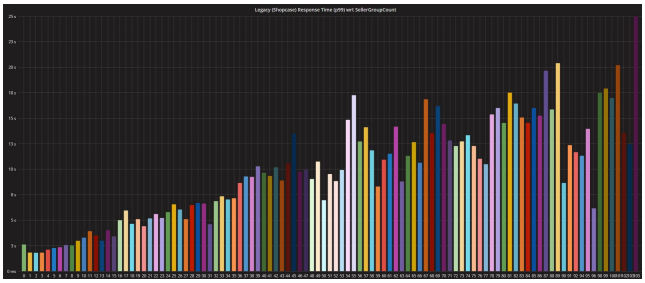

Figure 5: Legacy Shopping cart response time increasing linearly with increasing Seller Group count[1-100]

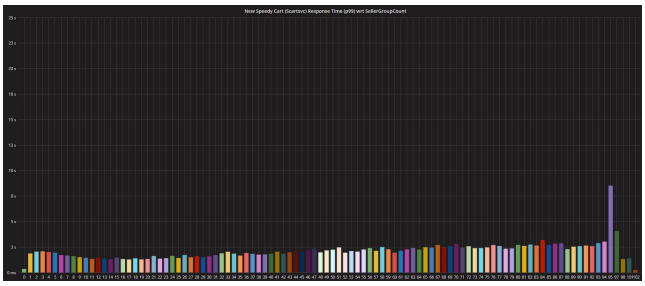

Figure 6: Optimized Shopping cart response time remains constant with increasing Seller Group count[1-100]

3.3 Real-Time Analytics

Visualizing the performance metrics of a service greatly helps in improving speed by identifying the bottlenecks in the system. Below we explain the tools we used to achieve this goal and validate the performance improvements.

- **Elasticsearch[5.2]** As a powerful search engine with aggregation capability, ES served as the perfect tool to store cart data and generate real time analytics.
- **Kibana[5.2]** Since the most popular tool to visualize and search data stored in ES is Kibana, we leveraged it to improve debuggability of shopping cart data.
- **Grafana** Although Kibana can be used for generating visualizations, its time series graphing capabilities did not fit our needs. We used Grafana primarily as a performance monitoring and measuring tool.

Furthermore, these tools played a major role in monitoring real-time errors and stability of the service while the service was being ramped.

4. RESULTS

The architectural changes along with optimizations, significantly improved cart performance. **Table 1** highlights the 99th percentile response time improvement of some of the key cart operations.

Table 1: Improvement in Shopping Cart Speed

Operation Type	99th Percentile		
	Old Cart	New cart	Improvement
Get	7.61s	1.48s	5X
Add	10.44s	4.65s	2X
Remove	6.40s	1.43s	4.4X

5. CONCLUSION

We have successfully launched a re-architected shopping cart which drastically improved the performance resulting in better user experience. The model we proposed incorporates a simplistic and elegant approach to build a scalable solution for such a complex system. This approach can be applied as an alternative solution to systems which have hierarchical design. The optimization techniques we described in this paper proved to be an efficient mechanism to overcome the bottlenecks caused due to heavy downstream services.

6. REFERENCES

[1] Elastic Search
https://www.elastic.co

[2] Kibana
https://www.elastic.co/guide/en/kibana/5.2/getting-started.html

[3] Grafana
http://docs.grafana.org/guides/getting_started/