Chapter 7
State Actor Model for Cloud-Based Online Auction

Yun Shu
Auckland University of Technology, New Zealand

Jian Yu
Auckland University of Technology, New Zealand

Wei Qi Yan
Auckland University of Technology, New Zealand

ABSTRACT

In recent decades, internet auctions have become the most significant e-commerce business model worldwide. With the rapid rise of cloud computing over the last few years, the legacy online auction platform is gradually being replaced using service-oriented cloud computing in real time. This chapter describes the design and implementation of a state and high-performance online auction system over cloud and proposes the methodology to provide persistent state records during the auction process so that we are able to ensure the reliability of submitted bid price and guarantee the security of price message in the delivery process. The authors employ actor-based applications to achieve stateful, parallel, and distributed architecture. Meanwhile, utilizing distributed databases provides secure and efficient data storage. To the best of the authors' knowledge, this is the first time that the actor framework has been applied to the online auction. The preliminary result is for implementation of high-performance and real-time bidding online auction.

DOI: 10.4018/978-1-5225-6313-6.ch007

INTRODUCTION

Cloud computing is becoming increasingly attractive to firms and developers nowadays. Compared with traditional Internet service provider, the legacy service providers are grouped into infrastructure vendors and service providers. The infrastructure providers (Platform as a Service) include Microsoft Azure, Amazon Web Services (AWS), and Google App Engine shown as Figure 1. The services provide impressive, reliable, and cost-efficient cloud-based platforms and lease these platform to the service providers who rent resources from infrastructure providers to serve end clients (Armbrust, 2010). Moreover, those developers who worked for the service providers have significant opportunities to transform their innovative ideas into highly scalable Internet services. We also call them Software as a Service (SaaS). These services can be easily expanded to a large scale to handle the expeditious increase in service demands (Armbrust, 2009).

Because of these features, we, therefore, have the opportunity to deploy actor-based applications in parallel and distributed system on cloud to overcome the bottlenecks of traditional client / server framework of the online auction. The actor model (Hewitt, P. Bishop, and R. Steiger, 1973) adopts an abstract concept to describe concurrency of the program that is centred on the actor unit which performs distributed computations and communicates through asynchronous information exchange. Thus, the actor is suitable for developing large-scale parallel programs. However, the concurrency of actors is limited by hardware resources and capability of logical computations (Agha, 1985). As the rapid progression and innovation of cloud technology in recent years, hardware constraints gradually weakened. The actor model is increasingly being applied to highly concurrent applications on SaaS, such as Microsoft Orleans and JVM Akka.

The main technical bottleneck of traditional online auction systems is that it is hard to handle significant amounts of data from different regions in a highly concurrent and parallel environment. More specifically, when an online auction system receives quotes from all over the world, traditional tree-tired architecture shown as Figure 4 will place cache area between middle tire and physical storage for improving I/O performance (Power and Li, 2010). Unfortunately, usage of the cache will directly lose the concurrency. The cache manager or application must adopt concurrency control policy to avoid deviations resulting from concurrent updates to a cached object (Strangers, 2005). Hence, the traditional architecture has to face major issues that come from traffic reliability and massive data processing within a short period. With or without the cache, this pattern cannot fulfil the requirement of conformity on a cache with rapid reaction for interactive access (Bernstein, et