When Client Server Model Meets Virtualisation

Terry Yin

October 1, 2014

Abstract

The client/server model and virtualisation are two perpendicular concepts in computer architecture. Both are used to separate the concerns in the system. This paper will discuss the similarity and differences between the two concepts, and will explore some real life examples of the combination of both design choices. It will also discuss the opportunity brought by virtualization to the startup business and telecommunication.

1 Comparing Client Server Model and Virtualisation

1.1 Similarity

Both the client/server model and virtualization are aimed to separate the concerns of the overall system (Dijkstra 1982, SoC). And "abstraction through **information hiding** is a primary factor in computer science progress and success," Colburn & Shute (2007). The abstraction sometimes leads to a modular structure like the client/server model, sometimes lead to a layered model like virtualisation.

1.2 Horizontal vs. Vertical separation in the system stack

The client/server and virtualisation architecture are two perpendicular concepts. They represent different ways of separation of concerns in the system design. The information hiding can be done horizontally or vertically (White et al. 2014, p.71-75).

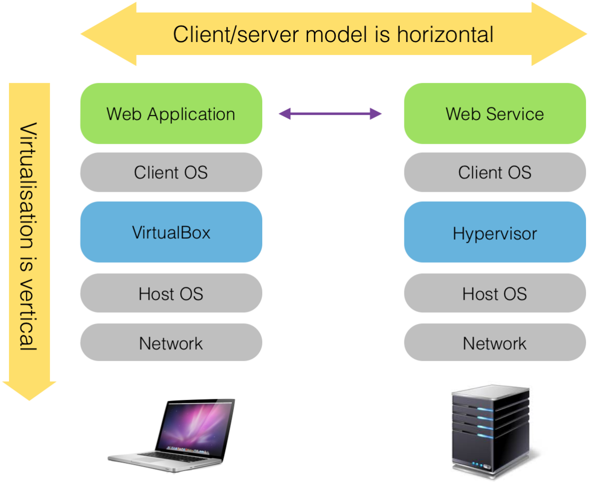

Figure 1: Example of Client/Server Model and Virtualization In Computer Architecture

1.3 Client/Server model is horizontal information hiding

A client/server model is an achitectural design decision on how to assign the *responsibilities* and how to *communicate* with each other in a distributed system. The result is a modularized structure with a centralized server and the distributed clients. From a system stack diagram, it looks horizontal.

An alternative archiyectural design on the same horizontal level is the peer-to-peer structure. Unlike client/server model, peer-to-peer structure is decentralized, and the responsibilities are distributed more equally among the peers. The design choices between client/server and peer-to-peer should be based on the requirement of the system.

One typical client/server architecture is the World Wide Web system. In the WWW system, a client browser is responsible for presenting the content, and the server is responsible for providing the content, and the communication is via HTTP protocol.

1.4 Virtualization is vertical information hiding

> "Abstractions should not depend upon details. Details should depend upon abstractions."
>
> Martin (1996)

Virtualization is to virtualize a dependency in the computer architecture. The motivation of virtualization is to have *Protected Variations* (Larman 2012, GRASP). From a system stack diagram, it looks vertical.

Information hiding is to protect a detailed implementation from it's changing dependencies. One example is software written in the Java programming language. We expect it to be written once and can be run on many different platforms. And that goal is approximately achieved by a JVM (Wikipedia 2014*c*, Java virtual machine).

Virtualization may happen in any layer in the stack, hardware, operating system, network, etc Smith & Nair (2005).

Despite the wonderful idea it sounds, there are still many concerns, like "Virtualization holds promise as a solution for many challenging problems. Expectations are running high. Can virtualization deliver?" KOTSOVINOS (2011). The expectations here include interoperability, scalability, fast delivery, easy troubleshooting, no silo, etc.

With some emerging solutions, like DockerWikipedia (2014*a*) and the cloud solutions, it looks more and more promising than ever in some areas.

2 Case studies

2.1 VPN, an example of both virtualization and Client/Server Model

A virtual private network (VPN) is a kind of computer network virtualization. It extends a private network across a public network. A VPN works in the client/server model. A client using VPN need to connect to a VPN server to setup a virtual network access to a private network. Wikipedia (2014*b*)

Because VPN is often used for security purpose, a centralized server makes the authorization and resource allocation easier.

2.2 Docker

"Docker is an open-source project that automates the deployment of applications inside software containers, by providing an additional layer of abstraction and automation of operating system–level virtualization on Linux."Wikipedia (2014*a*)

At the moment of writing this paper, Docker is still at its version 0.9, but the first official release should be coming soon.

The “containers” used by Docker is a metaphor from the containers on a docker ship. It’s the standardized and lightweight environment to contain software services. Software developers can develop the software locally and put the software in the containers to deliver. Then the containers can be shipped to any platform and run just as the developer expected, as long as the platform has the Docker engine.

The flexibility of delivery provided by Docker will be very important for small businesses, or in another term, the startup companies in the Internet industry in the next decades. Because, usually, the small business cannot afford to have their own server environment. (It’s a question whether any business should have their own server environment in the future.) They should be able to deliver quickly to any environment so that they are not limited by any vendor. Their ability to scale the service is also crucial to the small business. It will be a truly sadly story if their business opportunity suddenly shows up, but they cannot scale up to provide the service. With the virtualization provided by Docker and the cloud service, scaling becomes as simple as using docker to ship goods in containers from the workshop to the marketplace.

2.3 Virtualization for telecommunication infrastructure

The mobile network is probably the fastest developing computer network in the world, and it’s keep expanding. The number of mobile subscribers in the world is expected to be seven billion by 2016. This mean a large infrastructure behind it. Telecommunication technology is also developing very fast. New generation replaces old ones very quickly. Mobile operators have just done investment in the 3G network and now they have to switch to the 4G LTE network. The competition in the telecommunication makes the companies very cost sensitive. Jon Devlin (2013)

Telecom can benefit from virtualization in many areas.

The resource virtualization can balance the use of the telecommunication resource, increase the efficiency. The resource includes computing resource, bandwidth and radio network, etc.

The virtual operation systems can make the telecom equipments to take the advantage of the emerging multi-core processors and without giving up their legacy software system completely and rewrite everything. Jon Devlin (2013)

In the coming decades, the telecommunication operators and vendors with face even more competition than today. There are more players in this area, and there’s even open source telecommunication software and hardware emerging. Therefore, be cost-efficient is very important. Virtualization will surely be one of the directions they will need to look at.

3 Conclusion

Information hiding is essential in designing computer system. One way is to hide information horizontally, by assigning responsibilities to different roles in a distributed system. When one role is the centralized server, and another role is the client that request service from the server, we got a client/server model. There are other models when assigning responsibilities differently over the horizontal level. Usually, C/S model is simple and clear, but it cannot fulfill the need for highly dynamic distributed needs or high availability.

Another way is to hide information vertical, by virtualizing the dependencies of an upper level implementation. Virtualization removes the dependency on the detail, so that it protects the upper level from the variation of its dependency. Virtualization doesn’t come without cost, sometimes it makes the overall system performance optimization and troubleshooting very hard, and sometimes it leads to the speculative generality.

When making architectural design decision of a computer system, we need to consider carefully on both dimensions.

References

Colburn, T. & Shute, G. (2007), ‘Abstraction in computer science’, *Minds and Machines* **17**(2), 169–184.

Dijkstra, E. W. (1982), On the role of scientific thought, *in* ‘Selected Writings on Computing: A Personal Perspective’, Springer, pp. 60–66.

Jon Devlin, S. K. (2013), ‘cing telecom infrastructure capex and opex costs implementing advanced virtualization techniques’, *Avago Technologies* .
URL: *http://www.lsi.com/downloads/Public/noindex/LSI_WP_Virtualization_WindRiver.pdf*

KOTSOVINOS, E. (2011), ‘Virtualization: Blessing or curse?.’, *Communications of the ACM* **54**(1), 61 – 65.
URL: *http://search.ebscohost.com.ezproxy.liv.ac.uk/login.aspx?direct=true&db=bth&AN=56676283&site=eds-live&scope=site*

Larman, C. (2012), *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development, 3/e*, Pearson Education India.

Martin, R. C. (1996), ‘The dependency inversion principle’, *C++ Report* **8**(6), 61–66.

Smith, J. E. & Nair, R. (2005), ‘Introduction to virtual machines’, *Virtual machines: versatile platforms for systems and processes* pp. 9–10.

White, R., Morris, S. & Donohue, D. (2014), *The Art of Network Architecture: Business-Driven Design*, Pearson Education.

Wikipedia (2014*a*), ‘Docker (software) — wikipedia, the free encyclopedia’. [Online; accessed 30-September-2014].
URL: *http://en.wikipedia.org/w/index.php?title=Docker_(software)&oldid=627397448*

Wikipedia (2014*b*), ‘Virtual private network — wikipedia, the free encyclopedia’. [Online; accessed 30-September-2014].
URL: *http://en.wikipedia.org/w/index.php?title=Virtual_private_network&oldid=625807520*

Wikipedia (2014*c*), ‘Virtualization — wikipedia, the free encyclopedia’. [Online; accessed 1-October-2014].
URL: *http://en.wikipedia.org/w/index.php?title=Virtualization&oldid=626991943*