International Journal of Grid Distribution Computing
Vol. 8, No.5, (2015), pp.257-262
http://dx.doi.org/10.14257/ijgdc.2015.8.5.25

The Design of a Client-Cloud Computing Collaborative Model

Xu Guizhi

The Department of Computer Science, Jining Normal University, Wulanchabu, Inner Mongolia, 012000, China
Email: xu.guizhi@163.com

Abstract

As a new computing pattern, client-cloud computing can provide limitless computing and storage capacity according to the need for different types of client devices and avoid the increasingly skeptical cloud security question. For this purpose, a client-cloud computing collaborative architecture was built. On this basis, a collaborative service model based on the cloud and centered on clients was designed, and some crucial steps of collaborative application initiated by client were analyzed. A collaborative system development, maintenance method and process centered on clients were pointed out. The client-cloud computing collaborative model proposed in this paper provided a distributional, effective, independent, unfailing and secure architecture for the collaboration, highlighted the user's core position in collaborative activities, permitted individual users to start and manage collaborative application fast and easy from local.

Keywords: *Cloud computing, client, collaborative application, model design*

1. Introduction

The core idea of cloud computing is that all the information stored on the server can be called by anyone in any way, at any time and any place. Client-cloud computing is regarded as a trend of network development in the future. With the client as a core emphasizes the user's controlling shared information such as the music, the photo and the distributed files etc, even more complex application type including chat rooms and forums. Under this paradigm, the client that plays an increasingly important role which can connect peer-to-peer with other users and become the internet subject, in the case of the player that makes sharing music or video much easier, users can choose the favorite media to local player list and the distant friends can also add their media files to the playlist through authorized access. The player can play these songs or videos sequentially or randomly and all the users connected together can receive them at the same time.

Client-cloud computing that isn't a substitute but a useful supplement for cloud computing can let the user initiate transaction in the local and not overly relies on the functions provided by some organization or company website. It essentially provides more choices for users and easily builds environment sharing and communicating with others.

2. The Collaborative Architecture

The collaborative architecture consists of two parts: one is the service platform in the cloud extending the client service scope by the way of broker, the other is the client where the core application runs that can be dynamically loaded and adjusted when running. The clients including a series of abstract service components, together with the upper platform, form the complete collaborative space that is running and sharing.

The cloud is divided into three levels: at the bottom is the original cloud service system; the upper is the collaborative virtual platform that provides the virtual platform service for collaborative applications; the top is the shared agent environment that each client has on the collaborative virtual platform, through which the clients participate in the collaborative application collectively.

In the collaborative application, the users are collaborators and the core of the collaborative process, whose intention is expressed in the shared model by the collaborative application and modifies the model data.

The system structure is illustrated in Figure 1. When the user opens the collaborative environment, the client constructs the runtime environment firstly. During construction, the collaborative application list of interest performs initialization one by one, including registration to cloud and local loading. If an application is controlled completely by the client, all information of the application is loaded from the model warehouse of local and cloud according to setting. If the client is only participated in a collaborative application, the current sharing model of the application is loaded according to the URI and the model access strategy is set up according to its own allocated permission information. After the client runtime environment is constructed, users can open the loaded collaborative application and participate in the collaborative activities.

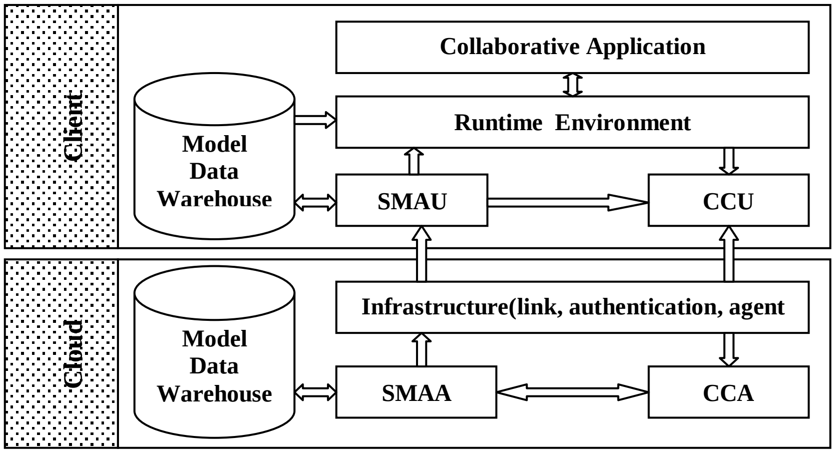

Figure 1. The Cloud Computing Collaborative System Structure

There are two types of exchange unit on the client side. One is Shared Model Access Unit called SMAU for short, which is responsible for receiving, sending sharing model data and handles synchronization according to the strategy. When received the change SMAU modifies the model data through running environment and informs the upper collaborative application in the form of events. Another is Collaboration Control Unit called CCU for short, which is responsible for concurrency and conflict arbitration during the collaborative process of application completely controlled by the client. Corresponding to the exchange unit, there are two types of agent on cloud: Shared Model Access Agent called SMAA for short and Collaboration Control Agent called CCA for short. The role of agent is similar to the corresponding unit that is mainly used for unit front. The purpose of the design is that agents have certain autonomy with the help of cloud model data warehouse.

The model data warehouse stores all state information of application which is located in the running client and the cloud collaborative virtual platform. It is to ensure the reliability and expansibility of the collaborative application that stored the state information in local and cloud respectively.

3. The Analysis of Collaborative Model

Analyzing the crucial steps of collaborative model can help expand on the design intention of framework. Follows are some crucial steps supported by the running client environment and cloud collaborative virtual platform, which can be called directly in different applications.

Step 1 the client registers to the cloud.

Registration is defined as the client makes offer of establishing agent environment to the cloud collaborative virtual platform for the use of collaborative function. After identity authentication of client, the cloud checks whether to satisfy the minimum resource request according to the authentication information. If it is satisfied, resources will be allocated and the agent environment is initialized, otherwise the registration fails.

Step 2 Initialize the running client environment.

The steps of initializing the running client environment rely on step 1. After the initialization of agent environment corresponding to user is completed, the cloud directs the client to download running environment constructed codes. The client dynamically loads the codes in the local and expands the running environment. The process is shown in figure 2.

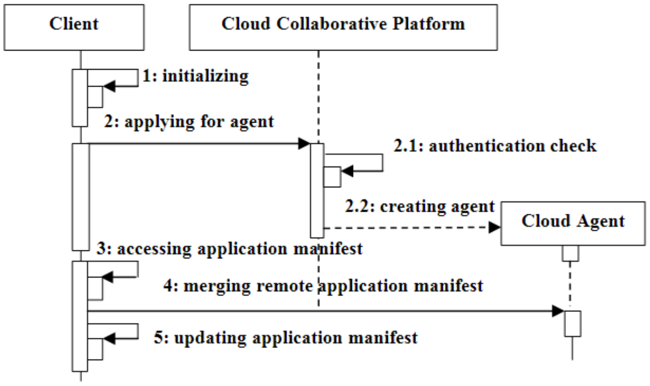
Client
Cloud Collaborative Platform
1: initializing
2: applying for agent
2.1: authentication check
2.2: creating agent
Cloud Agent
3: accessing application manifest
4: merging remote application manifest
5: updating application manifest

Figure 2. The Initializing Process of Runtime Environment

Step 3 Load the collaborative application.

Judge the application ownership first, that is whether the application is started by the current user. If the application is started by the current user, the managed property of application determines whether the application model data in the cloud agent warehouse is valid. Managed application is defined as the cloud agent maintains the lifetime of the collaborative application during self run time until the user takes over the application or timeout. If the client has managed before entering this time since last, the application model data downloaded from the cloud agent model database should combine with the local application model data, then be written to the local model database. If the application case isn't started by the current user, apply for joining the collaborative application to other agent environment according to the uniform resource identifier (Step 4).

Step 4 Join the collaborative application.

Each collaborative application has the uniform resource identifier, the default is HTTP protocol, that is composed of collaborative service domain name, cloud agent environment address and collaborative application alias or path. Joining the collaborative application must get access authorization of the collaborative agent environment firstly, after allowed to access, the client registers to the news release port of the agent environment, binds and continues to accept the update events related to the application, then starts to synchronize with the current view of the application on the whole(Step 5). When the synchronization is completed, the client renders the application environment locally and displays, in the end, notifies the other collaborative agent environment of its ready and can start receiving collaborative control command.

Step 5 Synchronize with the current view of the application.

Synchronizing collaborative application only after joined the collaborative application (Step 4). The application updates events that continuing to accept are recorded as queue A. The overall model data of the application current view trying to get is recorded as queue B. After acceptance, B determines the found location n in queue A where the last updated vector is. If n exists, it means that the synchronization is effective and the client has kept up with the updated rhythm of the application. Then B is saved locally, beginning from location n+1 in queue A the updated vector is executed in sequence to the local model, and the synchronization is completed. If n doesn’t exist, the updated vector may be lost between A and B, then the collaborative application synchronizing steps should be re-executed and to remind the users.

Step 6 Exit the collaborative application (temporary).

Temporary exit may be the user’s active behavior or other collectively known as passive exit caused by power failure, network link lost etc. According to the ownership of collaborative application, temporary exit is divided into collaborative participating user’s and collaborative application owner’s. When the collaborative participating user exits actively, the binding behavior for the application event should be canceled. The collaborative participating user may exit passively; the agent environment will detect the survival of heartbeat regularly to all the participated clients, and cancel the event subscription when judging a client is lost.

Step 7 the owner exits the collaborative application permanently.

Permanent exit only for the owner of the collaborative application means the ownership is transferred (Step 8) or termination (Step 9).

When exiting permanently, prompted information whether to save the application model data appears.

Step 8 Transfer the ownership of the collaborative application.

The owner of the current application instance chooses transferred user B from the collaborative participated user list and makes the transfer request. If the other chooses agreement: the current agent environment at first stops receiving the change operation submitted to the application collaborative control agent, does with all the changed operating vector received currently and delivers in the form of events to all other clients who subscribe the application, at last contains a special event PAUSE and makes other clients waiting; the agent environment A sends all the latest application model data in the warehouse to B; after received application model data, B should verify integrity and rebuild the uniform resource descriptor i of the application which is returned to agent environment A and gets a receipt; B executes step 6 to exit the collaborative application; B executes step 3 to load the collaborative application and generate a new instance; after received the new generated resource descriptor, A sends redirected special events REDIRECT i to all clients who subscribe the application; after received redirected events, other client updates resource descriptor of the application in its list to i, and executes step 4 to join new instance of the collaborative application.

Step 9 End the collaborative application.

Different from the ownership transferred behavior of collaborative application, the ended behavior should be designed cautiously. The ending step defines a default back, so the collaborative application owner has the chance to cancel ending countdown and the other clients participated in collaborative application can have chance to backup its visual model data.

Step 10 Exit the collaborative agent environment.

Iterate application list, execute step 6 for each to exit collaborative application. When all application exits is finished, release the agent environment.

4. Conclusion

While providing local security the client-cloud computing can use the advantages of cloud computing about reliability, distribution etc. to combine with the user's independent, personalized service will. The advantages of the client-cloud computing collaborative model researched in this paper are as follows.

4.1. High Reliability

Due to the client oriented model, the core storage actually occurs in the user's local where users can archive and copy data by themselves. In addition, the cloud agent also provides remote storage for collaborative data that further increases the capacity against the risk of data loss. Connecting services provided by cloud computing platform ensure the stability of transmission and the reliability of platform.

4.2. High Autonomy

The collaborative system for cloud environment presented in this paper has many autonomy: users can select cloud services themselves or switch; users can deploy, operate or disposal collaborative application; users can administrate collaborative data and collaborative activities; users can invite other users to participate in collaboration and give them basic rights; users can apply to other interested collaborative application case for joining; users can let cloud agent environment take over the collaborative application for self operation; users can leave or return to the collaborative environment arbitrarily and so on.

4.3. Good Commercial Prospects and Social Value

Take advantage of the network connection service, collaborative virtual platform built on a cloud platform can bring new service products for the cloud service provider. Individual users can build their own collaborative virtual platform and pay the cloud service provider reasonably for demand. Because the collaborative virtual platform has cloud irrelevancy, individual users can choose different cloud service provider.

When the barrier to start collaboration is lower and participating in collaboration is simpler, communication among people will become more extensive and more efficient and finally drive its own development.

References

[1] Y. H. Yu, L. Ning and W. Z. Liu, "Combining HTML 5 with MVC Framework to Simplify Real-Time Collaboration for Web Development", 1st International Conference on Multimedia Information Networking and Security, **(2009)** February 29-32, Washington, USA.

[2] Y. H. Yu, L. Ning and W. Z. Liu, "A Web Collaborative Framework Based on HTML5", Microelectronics and Computer, vol. 11, **(2012)**.

[3] P. Lubbers and R. Greco, "HTML5 Web Sockets: A quantum leap in scalability for the web", JayView, vol. 21, no. 1, **(2010)**, pp. 3-7.

[4] W3C. HTML5 Web Messaging. W3C Working Draft 17 March 2011. http://www. w3.org/TR/webmessaging/

[5] M. Armbrust, A. Fox and R. Griffith, “Above the clouds, A Berkeley view of cloud computing”, Technical Report ECS-2009-28, **(2009)** University of California, Berkeley, CA.

[6] L. Qian, Z. G. Luo and Y. J. Du, “Cloud Computing, An Overview. Proceedings of the 1st International Conference on Cloud Computing”, **(2009)**, Heidelberg, Berlin, German.

[7] K. Birman, G. Chockler and R. Renesse, “Toward a cloud computing research agenda”, ACM SIGACT News, vol. 40, no. 2, **(2009)**, pp. 68-80.

[8] P. C. Shih, D. H. Nguyen and S. H. Hirano, “GroupMind, supporting idea generation through a collaborative mind-mapping tool”, Proceedings of ACM GROUP '09 international conference on supporting group work, **(2009)** New York, USA.

Author

XU Guizhi, (1970), female, associate professor of computer science, The Department of Computer Science, Jining Normal University, Inner Mongolia, Wulanchabu, P.R.China