Towards an Emerging Cloudware Paradigm for Transparent Computing

Wei Wang[+] Jingxuan Zhang[+] Dong Guo[+×] Qiao Xiang[+#] Chenxi Huang[+]
Jinda Chang[+] Liqing Zhang[+]

+Tongji University #Yale University ×Cloudware Labs

ABSTRACT

Transparent computing is an implementation of ubiquitous computing that is aimed at providing active services for users. In transparent computing, the execution (computation) of computer instructions and data is temporally and spatially separated from their storage. Cloud computing solves the issue of data cloudlization, while transparent computing solves the one of software cloudlization. This paper define the concept of *Cloudware*, and discusses how to deploy Cloudware in cloud environment efficiently. Based on a loosely coupled von Neumann computing model (also called Cygnus Model), we proposes a new platform to construct the PaaS platform which can directly deploy software on the cloud without any modification, while achieving a new model by the browser services, and we call it *transparent computing 2.0*. By using micro-service architecture, we can achieve such characteristics as good performance, scalable deployment, faults tolerance and flexible configuration. Finally, we presents a Cloudware PaaS platform prototype, called *CloudwareHub*, and demonstrates the promising of the transparent computing 2.0 with the support of Cloudware technology.

Keywords

Cloud computing; Cloudware; Transparent computing; Service

1. INTRODUCTION

Transparent computing is a computing paradigm that is aimed at providing active services for users [1]. That is to say, users do not need to be aware or well informed of the minute details of the technology applied in the system; instead, they need to only care about the services they want, and the quality of the services. In order to achieve this goal, a transparent computing based system consists of a network server and one or more client terminal. The client terminal is rather simple and light-weighted, almost like a bare computer. It only stores the underlying BIOS and a small fraction of protocol and management program. On contrast, OSs, applications and user data are all regarded as software resources and stored in the network server.

On the other side, the rapid development of cloud computing and Internet technology not only promotes the progress of computer software and hardware, but also makes people change the way of using software. The concept of cloud computing makes the Internet technology reach its climax, and a lot of traditional industries are in the transition to the Internet mode to improve social efficiency, and further affecting every people's work and life. In this environment, the traditional software deployment paradigm will also requires a transition.

UCC 2016, 6-9 December 2016, Shanghai, China
Copyright 2016 ACM ISBN 978-1-4503-4616-0 …$15.00.
http://dx.doi.org/10.1145/2996890.2996908

First, the idea of IT services as resources is becoming more and more popular, showing the trend of everything as a service (XaaS). Users can easily enjoy the different levels of software as a service through networks, and such services have become the essence and core concept of cloud computing. IaaS, PaaS and SaaS are some representative service models, which have been widely studied and deployed. At the same time, with the continuous deepening of virtualization and container technology, e.g., Docker, on behalf of the containers, gradually infiltrated into cloud computing at all levels of the system from the development to the operation and maintenance of the whole process. The micro-services architecture has become an architectural style and design patterns. It advocates the mode to split the application into a series of small services, in which each service focuses on a single business function, running in a separate process, thus making the service operating with clear boundaries. Light communication mechanisms can also be adopted, such as HTTP/RESTful, to meet the needs of business and users. Micro-service, as an architecture model of transformation, is not accidental. It is a reflection of the architecture model, development, operation, and the maintenance methodology.

With the continuous optimization of the network environment, especially the rising of 5G related wireless communication technology, it is easier for users to gain high-speed access to the Internet, making it easy for traditional software gradually migrating to the cloud. For end users, the browser is the main entrance to the Internet and browser technology is also in rapid development, from the simple analysis of the HTML file to the new HTML 5, CSS 3 and Web OS etc. This provides a solid foundation for software in cloud migration and web access. Access to software services through the browser will be an important direction of future software development, and the cloudlization of software will also become one of the important forms of software in the future [2].

In summary, cloud computing provides environment for micro-services to build a software and the runtime environment. Meanwhile, utilizing the advanced Internet technology to realize web-based software will be the direction of software development and the tendency. In the cloud, the software will no longer be a simple code entity, but a series of services delivered to the users through the Internet. In this paper, we refer this paradigm of Internet plus software as Cloudware. We believe it will become the main form of future software paradigm. At the same time, we refer to this Cloudware based computing model as transparent computing 2.0.

In order to verify the feasibility of transparent computing 2.0 and Cloudware, this paper proposes and develops a PaaS platform based on the micro-service architecture and container technology.

In this platform, the traditional software can be directly deployed on the cloud, and users can connect to the cloud by using the browser to interact with the Cloudware.

2. RELATED WORK

Transparent computing was firstly proposed by Yaoxue Zhang in 2006, and then, his team achieved a series research results in terminal and Meta OS [3], TransOS [4] , and file getting middleware [5]. With the profound changes in cloud computing, big data, networking and other technology, transparent computing theory has started to show more and more broad prospects. On the other side, as the Web advent as an Internet portal, browsers have become a kind of indispensable tool to our lives. More and more people use browsers perform commercial activities and access business applications on the Internet. As working with a browser continues to gain acceptance, an emerging strategy for software providers is to publish their applications online. This is so called Desktop Software Virtualization (DSV). DSV can be regarded as one of the implementation of transparent computing.

Researchers have attempted to realize this DSV concept in several ways. One solution is to create a Web user interface (UI) for a desktop application and providing a wrapper to mediate the interaction between the UI and the application [6]. However, software companies would need to implement a dedicated Web UI and wrapper for each desktop application. A simpler solution is to design a library for desktop applications to generate Web UIs dynamically [7]. This provides a generic way to publish desktop applications on the Web, but there’s little support for managing the execution of applications. Other research has deployed applications in a desktop environment and adopted a Web-based remote desktop method to support users in interacting with the applications through a browser [8]. Although users can manage applications directly in a desktop environment, sharing the whole desktop is inefficient, especially in a network environment with low bandwidth. In recently, Chen and Hsu [9] introduced a desktop application service (DAS) framework that lets a Web server provide services based on existing desktop applications. But, in essential, this method is still a virtual desktop based methodology, and cannot utilize the resource efficiently, such as underline computing and storage.

To solve the problems above, this paper proposed a Cloudware paradigm based method which can directly deploy software on the cloud. With the emerging of the Cloudware, transparent computing will further evolve to its next generation which will be described in this paper in detail.

3. BASIC CONCEPT OF CLOUDWARE

Software are everywhere nowadays. With the development of cloud computing and virtualization, more and more software tend to be deployed on cloud, making it free for the local resource. This is actually a form of service, and we call this diagram of software Cloudware. It is a specific form of SaaS, but the idea is different in that Cloudware relocates the local OS and runtime environment to the cloud without modification, and the users can communicate with it through a unified interacted platform.

Cloudware is a software paradigm suiting for scalability and on-demand service under cloud environment, which makes direct transportation of software in the cloud in a convenient way, and end users can get access to it through any browsers [10][11].

From the above Cloudware concept, you can see that transparent computing will also evolve to a new one which we call it transparent computing 2.0 (short for TC 2.0).

To be specific, Cloudware has the features such as: running on the cloud, flexible resource configuration, rendering on the cloud, fast boot, interaction with the Internet, unified platform, transparent communication, et al.

With the rising of the container technology, GPU virtualization technology, 5G network technology and HTML 5 front-end interactive technology, achieving such characteristics becomes possible. In particular, the characteristics of Cloudware depends on the following related technologies, such as virtualization, cloud interactive rendering, container, media stream data compression, et al.

Cloudware can be regarded as an emerging software paradigm for cloud computing [11]. A software paradigm usually concerns four aspects [2]: what is to be constructed and executed; how to develop the resulted software artifacts or entities (development techniques); how to run the artifacts or entities (runtime system supports); and how well the constructed and executed software can perform (the promised software qualities).

Cloudware software model. The Cloudware model should specify the form, structure and behavior of the software entity as well as the user interaction. These will determine the principles and features of the corresponding software technologies (programming languages, development approaches and runtime mechanisms). The Cloudware software model leverage both legacy software and new features. The basic Cloudware can be built upon current popular technologies, such as object-oriented technologies or services computing technologies. But new capabilities should also be provided. The Cloudware software model concerns three aspects: client, cloud and interaction channel, as well as their relationships.

Cloudware operating platform. Software operating platform realizes the elements and their relationships of the software model. Cloudware operating platform provide a runtime space to operate Cloudware entities and their interaction. It should equip legacy software systems with Cloudware features, and should also manage the applications and itself in a more intelligent and automatic manner.

Cloudware engineering approach. The engineering approach should systematically control the whole lifecycle of Cloudware development, including requirements specification, design, implementation, deployment, maintenance and evolution. The Cloudware engineering approach follows the core principle of *Software Architecture of the Whole Lifecycle*. Software architecture acts as the blueprint and controls every stage of Cloudware development. To support the online development of Cloudware applications, the Cloudware and their on-demand interaction are implemented and governed based on software architecture.

Cloudware quality assurance. Software on the Cloud usually serves a large number of users in an online and simultaneous style via Internet. Cloudware quality framework should not only define quantitative and qualitative measurement methods for various quality attributes such as performance, reliability and usability, but also make comprehensive tradeoffs among these attributes. To promise the Cloudware quality, it requires the quality assurance mechanisms by both engineering approach at development time (e.g., testing, verification and validation) and software running at runtime (e.g., online evolution, autonomic system management).

As Cloudware paradigm emerges at it's very early age, we mainly focus on its performance aspect of the quality, such as Cloudware boot time, networking low latency, and user experience in this paper. The research of the whole quality assurance mechanism for Cloudware will be carried out in the near future.

From the above discussion, traditional cloud computing can only achieve remote data storage, while transparent computing 1.0 realized the server of the data, the remote software and even the operating system and the computing on the client. The transparent computing 2.0 further achieve a whole cloud of data, software, operating system and computing, displaying them through client.

The difference between transparent computing 1.0 and transparent computing 2.0 can be shown in Table 1. In addition, we can also compares the differences between the Cloudware and desktop application as well as the cloud desktop, as shown in Table 2 and Table 3 separately.

Table 1 The difference between TC 1.0 and TC 2.0

	TC 1.0	**TC2.0**
Data storage	Server	Cloud
Application storage	Server	Cloud
Application running	Client	Cloud
OS storage	Server	Cloud
OS running	Client	Cloud
Interaction	Client	Client

Table 2 Differences between Cloudware and Web application

	Cloudware	**Web application**
Computing	Completely on cloud	Partialy on terminal
Output	Windows interactive images	HTML, CSS, Image
Input	Mouse and keyboard event	HTTP request
Storage	Copletely on cloud	Partialy on terminal
Recovery	Yes	No

Table 3 Differences between Cloudware and cloud desktop

	Cloudware	**Cloud desktop**
Computing	Completely on cloud	Partialy on terminal
Service type	Only sofware service	Desktop system service
Install	Immediately used	Download and install
Output	Windows interactive images	VRAM rendering information

4. CYGNUS MODEL FOR TRANSPARENT COMPUTING 2.0

Since the computer science appeared, the scientists represented by John von Neumann dominates the main stream of computing paradigm, and these theories are called von Neumann Architecture which is used by the majority of computer architecture researchers. It indicates that computer is made by five components, namely storage, operator, controller, input and output device, as shown in Figure 1 (left). We can see a tight relationship between these five components, and all of these parts are necessary.

The traditional von Neumann architecture can be represented by the following tuple:

$$VC = (C, A, M, I, O) \quad C, A, M, I, O \neq \phi \qquad (1)$$

in which C represents for the controller, A for operator, M for storage, I for input, and O for output.

On the other side, in the transparent computing 2.0 computing model, the input, output, storage and computing do not locate in a single computer system. They may be in different areas and be connected through the Internet. For examples, the input and output for cloud are deployed on the terminal while storage, controlling and computing on the cloud, and they communicate with each other through networks. And this article calls this kind of model the Loosely Coupled von Neumann computing model (also called *Cygnus Model*, as its shape like the constellation of Cygnus), which is the basic theory of transparent computing 2.0.

If we investigate the computing model from an instruction view (instruction processing, instruction storing and result presenting), we can see that traditional PC tightly coupled these three part into a single monolithic machine, while transparent computing 1.0 and the earlier C/S separate the instruction storing and instruction processing together with result presenting into different physical machines. In the age of cloud computing, with the support of Cloudware paradigm, transparent computing 2.0 totally separate the three part into three kinds of physical machines, which can provide deferent service in a elastic on-demand way. Figure 2 to Figure 4 show these different computing model from a computing instruction view, which shows the essential difference of these computing models.

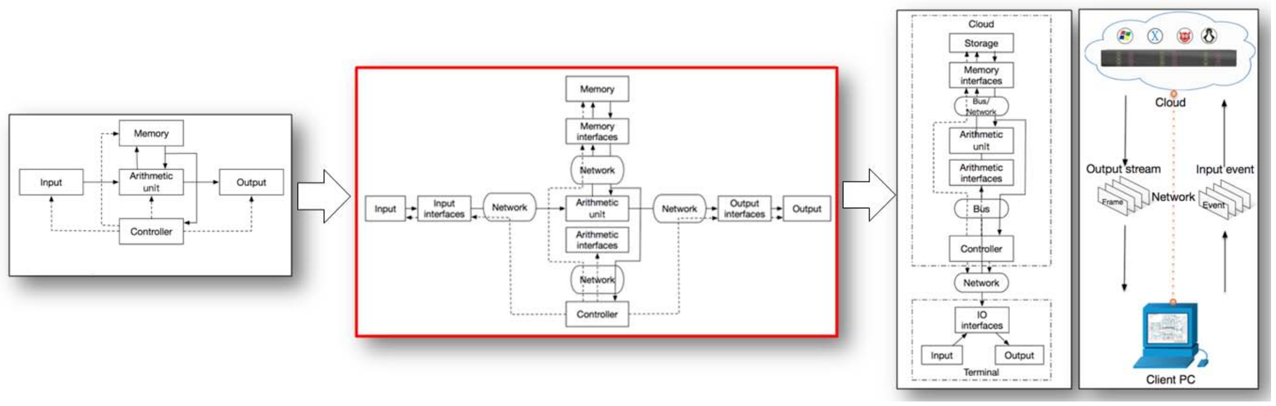


Fig. 1 von Neumann Architecture (left), Cygnus model (middle) and its application for transparent computing 2.0 (right)

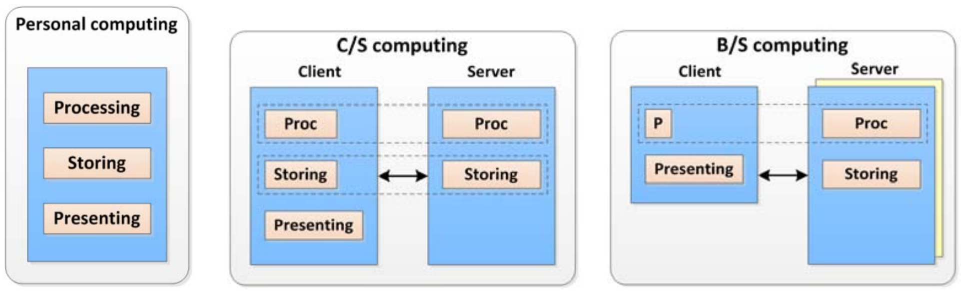


Fig.2 Traditional tightly coupled von Neumann model for PC (left) , C/S computing model (middle) and B/S computing model (right)

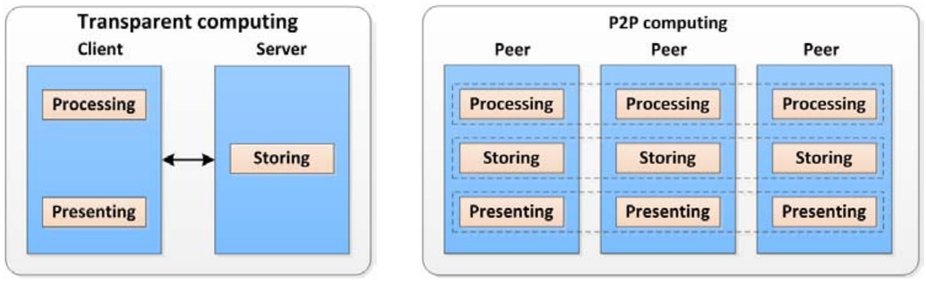


Fig.3 Transparent computing 1.0 model (left) and P2P computing model (right)

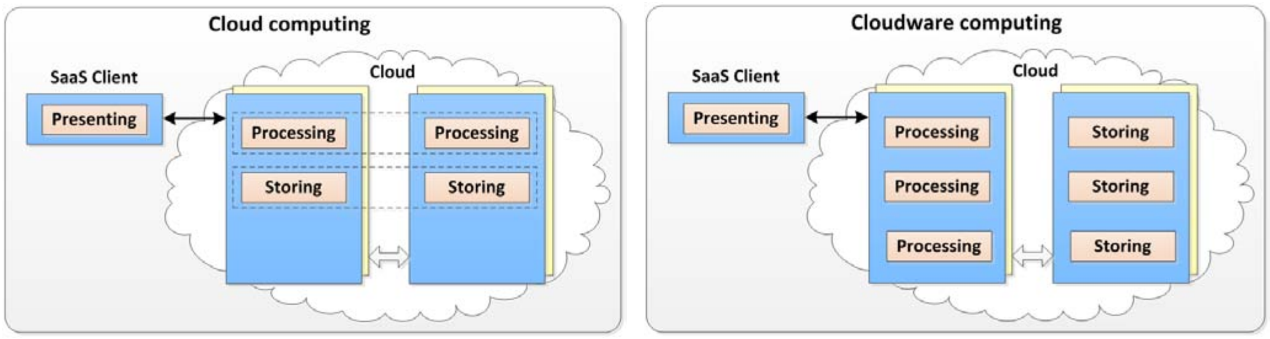


Fig.4 Cloud computing model(left) and transparent computing 2.0 model (right)

4.1 Cygnus Model Basic

The key Cygnus model is to decouple the five components, and to make them into complete systems which can run separately and can communicate with each other through networks. A typical Cygnus model is shown in Figure 1 (left) (solid line represents for data stream while dotted line for controlling stream).

We can see that input, output, computing, storage and controlling are all connected by networks. Their external communication is achieved by correspondent interface. This makes it possible for those components to extend themselves.

The Cygnus model can be represented by the following tuple:

$$LVC = (C, A, Ai, M, Mi, I, Ii, O, Oi, N) \quad C, A, Ai, M, Mi, N \neq \phi \quad (2)$$

in which C represents for controller, A for operator, Ai for the calculating interface, M for storage, Mi for storage interface, I for input, Ii for Input interface, O for output, Oi for output interface, N for network. We can see that input and output are not necessary, which can be immediately achieved through networks.

In addition, the network devices include those physics connection or wireless connection devices which support Internet or other bus architecture, not only IP network. N can be represented as a the following tuple:

$$N = (Wire, Wireless, Routed) \quad (3)$$

in which *Wire* represents for the physics connection; *Wireless* for the wireless connection like LAN; *Routed* for those devices who can communicate with indirect passed route like WAN.

In the Cygnus model, Interface is the key for the communication among different components to achieve the protocol in distinct level. In the Cloudware system, different components are abstracted into services while Interface becomes the interface of the external services.

4.2 The features of Cygnus model

The module features of Cygnus model are very typical, endowing every component with high flexibility, which are listed as below:

Infinite extension of component through network. Because the network introduced in every component of Cygnus model, it is easy for the model to extend to the whole network.

The independent component system. For the single component in Cygnus model, it must be able to run separately and to provide external interface, ensuring its validity even in the case that external component failed to work.

The hot plug components. The Cygnus model supports hot plug. For instance, any display devices with different resolution can be immediately replaced with no break during the process.

Infinite storage. Because the storage is deployed behind the interface, the physics storage is not transparent to the computer system. Therefore the logical storage information can only be accessed by the storage interface. It is easy to establish the backend distributed storage in this sense, namely the infinite storage space.

Networking self-adaption. The networking in Cygnus model does not refer to the present IP network, but the network devices of physics or wireless connection. The protocols in the lower level dependent on the unified controller.

4.3 Cygnus model for TC 2.0

The typical features of transparent computing 2.0 reflects the Cygnus model, the computing, storage and controlling are deployed on the cloud while the input and output on the terminal. The communications between the components are achieved by network， and the input and output are hot-pluggable, as shown in Figure 1 (right).

Transparent computing 2.0 model uses terminal environment to achieve the independent system of input and output, and to interact with each other through TCP(UDP)/IP protocol. The programs on cloud can utilize the storage resource, and they can also use the system call of OS to achieve storage function. That of the computing component is supported by the CPU of the Physics machine, and that of the controller is supported by the correspondent with the related Cloudware application. In transparent computing 2.0 computing model, every module is abstracted as service, and the interactive service of Cloudware is deployed on terminal while the computing and storage on the cloud.

5. IMPLEMENTATION AND DEMOS

In order to verify the transparent computing 2.0 technical conditions on the current cloud, we developed a Cloudware of PaaS platform, called CloudwareHub, and demonstrates it in this section.

5.1 Architecture

CloudwareHub is a platform for the Cloudware development, testing, deployment and maintenance, both for developers and users. The Cloudware PaaS platform overview architecture is shown in Figure 5. From Figure 5, we can see that three main parts make up the whole Cloudware PaaS platform architecture: Cloudware server pool, load balance server, and NAT server pool.

Due to the adoption of the micro-service architecture, the CloudwareHub system decomposes a single application into a set of focused services, each of which is according to a component with the function of independent compilation, deployment and extension. With the support of Cloudware PaaS platform, transparent computing 2.0 has some advantages when compared with the single architecture:

The limited complexity. The decomposition in the application reduces the complexity of the original endless accumulation. Each micro-services focus on a single function, and articulate the service boundary through well-defined interfaces.

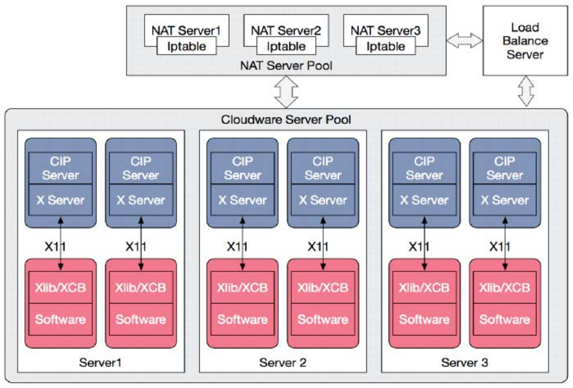


Fig. 5 The overall architecture of Cloudware

Independent deployment. Since the micro-services have an independent running processes, each micro-services can also be deployed independently. When a cloud service are changed without compiling, there is no need to deploy the entire application.

Flexible technology selection. By micro-services architecture, technology selection is decentralized. Each team can make their decision according to the status and development of its own service industry, having the freedom to choose the most suitable tools for the development.

Faults tolerance. When a failure occurs in the traditional architecture of a single process, it is likely result in diffusion, making the global application crashed. In the micro-services architecture, the fault will be isolated in a single service.

Scalability. Single application architecture can achieve scale by copying the entire application to a different node. When different components of Cloudware elements are different in expansion needs, micro-service architecture will reflect its flexibility because each Cloudware can be expanded independently

5.2 Running environment and demonstration

CloudwareHub itself provide micro services environment. To achieve scalable deployment, faults tolerance and flexible configuration, we base our system on IaaS. To test the validity, we make our CloudwareHub prototype system run on two cloud hosts with two cores, 4GB memory and 20GB disk. Web Service runs on one of them and can be accessed by www.cloudwarehub.com.

Container Service is based on Linux with Ubuntu Server 14.04, Docker tools, Xorg devices and programming libraries. Parts of Cloudware brokers consist of C manager and X manager, managing the Xorg and Docker internally and providing TCP externally. All the systems above are deployed on Ucloud (www.ucloud.cn), offering Cloudware service.

On CloudwareHub platform, you can first find a specific Cloudware in the Cloudware base, as shown in Figure 6, and then you can add your Cloudware in your workspace. Finally, you can run your Cloudware from your workspace by double clicking the Cloudware icons, as shown in Figure 7 (running a Matlab Cloudware service). We also developed a Web desk style environment for the Cloudware service, named OpenWebDesk. Users can use the OpenWebDesk environment like traditional desktop, as shown in Figure 8, with the support of a Cloudware management system with can manage the whole lifecycle of the Cloudware service, as shown in Figure 9.

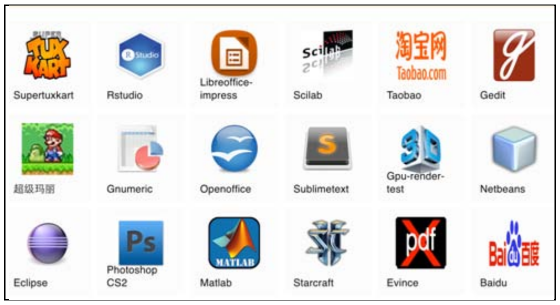


Fig. 6 Find Cloudware in the Cloudware base

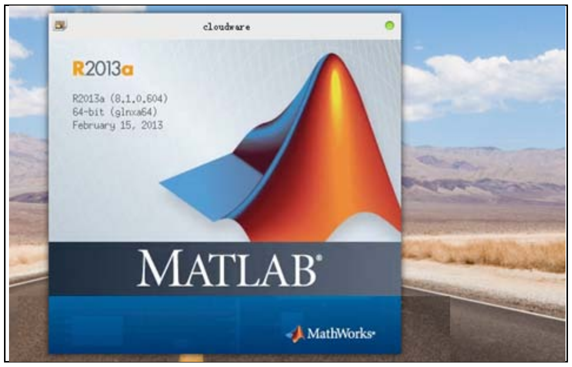


Fig. 7 Ruining a Matlab Cloudware service

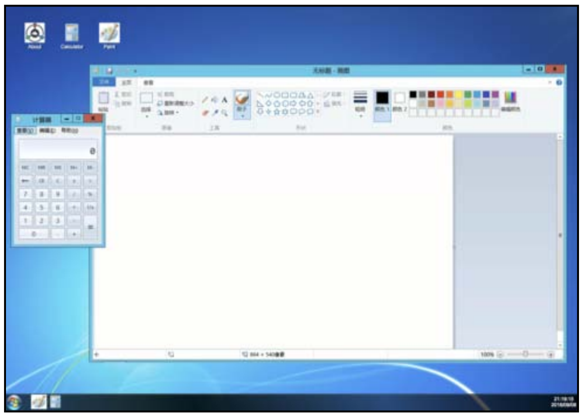
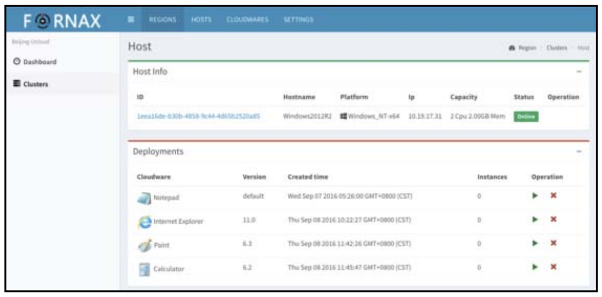


Fig. 8 OpenWebDesk environment



Host

Host Info

ID	Hostname	Platform	Ip	Capacity	Status	Operation
1eea16de-b30b-4858-9c44-4d65b2520a85	Windows2012R2	Windows_NT-x64	10.19.17.31	2 Cpu 2.00GB Mem	Online	

Deployments

Cloudware	Version	Created time	Instances	Operation
Notepad	default	Wed Sep 07 2016 05:26:00 GMT+0800 (CST)	0	▶ ✖
Internet Explorer	11.0	Thu Sep 08 2016 10:22:27 GMT+0800 (CST)	0	▶ ✖
Paint	6.3	Thu Sep 08 2016 11:42:26 GMT+0800 (CST)	0	▶ ✖
Calculator	6.2	Thu Sep 08 2016 11:45:47 GMT+0800 (CST)	0	▶ ✖

Fig. 9 Cloudware management system

6. CONCLUSION

With the mature of Cloud computing, how traditional software can utilize cloud platform, and give the cloud service to the client is a hot research problem. In this paper, we proposed a new Cloudware PaaS platform based on micro-service architecture and light weighted container technology. The traditional software can be directly deployed in this platform without modification, and provide service to the client by a browser, which we call it transparent computing 2.0. The idea of Cloudware computing model and transparent computing 2.0 are on their early state, and much work have to do before it can put into practice. Under present circumstance, even though the containers and the video compression can meet the need of the majority of users, technological challenges also exist including further using rate increase, video stream cross-compression, self-adaptive network compression, real time network, interactive terminal unified platform, GPU virtualization and transparent storage technology, etc. We will continue to focus on researches and improvements in the related fields.

7. ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (Grant NO. 61672384), the Program of Shanghai Academic/Technology Research Leader (15XD1503600), the Open Project Program of the State Key Laboratory of Computer Architecture (Institute of Computing Technology, Chinese Academy of Sciences, CARCH201408).

8. REFERENCES

[1] Y.Z. Zhou, Y.X. Zhang. Transparent Computing: Concepts, Architecture, and Implementation. CENGAGE Learning Press, 2010.

[2] H. Mei, G. Huang, T. Xie, Internetware: A Software Paradigm for Internet Computing, IEEE Computer, 2012, 45(6): 42-47.

[3] Y.X. Zhang, Y.Z. Zhou, 4VP: a Novel Meta OS Approach for Streaming Programs in Ubiquitous Vomputing, In proceedings of 21st International Conference on Advanced Information Networking and Applications, 2007, AINA'07, IEEE, 2007, pp.394–403.

[4] Y.X. Zhang, Y.Z. Zhou, TransOS: A Transparent Computing-based Operating System for the Cloud. International Journal of Cloud Comput, 2012, 1(4): 287–301.

[5] K. Guoa, Y. Tanga, et al., Optimized Dependent File Fetch Middleware in Transparent Computing Platform, Future Generation Computer Systems, 2015.

[6] A. De Lucia et al., Developing Legacy System Migration Methods and Tools for Technology Transfer, Software: Practice and Experience, vol. 38, no. 13, 2008, pp. 1333-1364.

[7] Z. Karampaglis et al., Secure Migration of Legacy Applications to the Web, Information Technology and Open Source Applications for Education, Innovation, and Sustainability, Springer, 2014, pp. 229–243.

[8] S. T. Wang et al., Development of Web-Based Remote Desktop to Provide Adaptive User Interfaces in Cloud Platform, Int'l J. Computer, Information, Systems and Control Eng., 2014, 8(8): 1195-1199.

[9] B. Chen, H. Hsu, Y. Huang, Bringing Desktop Applications to the Web, IT Professional, 2016, 18(1): 34-40.

[10] D. Guo, W. Wang, J.X. Zhang, et al., Towards Cloudware Paradigm for Cloud Computing, In Proceedings of The 9th IEEE International Conference on Cloud Computing (CLOUD), 2016, San Francisco, USA, June 27 - July 2, 2016.

[11] D. Guo, W. Wang, J.X. Zhang, et al., Cloudware: An Emerging Software Paradigm for Cloud Computing, In Proceedings of The Internetware '16, September 18, 2016, Beijing, China.