2016 IEEE International Conferences on Big Data and Cloud Computing (BDCloud), Social Computing and Networking (SocialCom), Sustainable Computing and Communications (SustainCom)

Integrating SaaS and SaaP with Dew Computing

Yingwei Wang David LeBlanc
School of Mathematical and Computational Sciences
University of Prince Edward Island
Charlottetown, Prince Edward Island, Canada
Email: {ywang, dcleblanc}@upei.ca

***Abstract*—Software as a Service (SaaS) is one of the major service models of cloud computing and a new software delivery model; in this software delivery model, software is hosted centrally, is licensed to users on subscription basis, and typically is accessed by users through a Web browser. Before SaaS was introduced, the traditional software delivery model was Software as a Product (SaaP): software was sold to users and installed in users local computers. In this paper, we discuss the advantages and disadvantages of SaaS and SaaP, and suggest a new software delivery model that is an integration of SaaS and SaaP. Dew computing can be used to integrate SaaS and SaaP. This new software delivery model has some great features.**

1. Introduction

Software as a Service (SaaS) [1] [2] is one of the major service models of cloud computing along with Platform as a Service (PaaS), Infrastructure as a Service (IaaS) , and other service models. SaaS is not only a service model of cloud computing, but also is a software delivery model. In SaaS, software is hosted centrally, licensed to users on subscription basis, and typically is accessed by users through a Web browser.

In an SaaS solution, Web-based software is hosted exclusively by the vendor. The vendor furnishes the users with access to the software in accordance with agreed-upon standards. All a user needs to access the software is an Internet connection. Examples of SaaS solutions include Hotmail, Salesforce, and Facebook.

SaaS is an alternative to the more established, yet increasingly less popular traditional software delivery model by which software is sold to users as a product and installed in users' local computers. Before SaaS was introduced, this traditional software delivery model played a dominant role. There is no widely-accepted terminology associated with this model. For convenience sake, we use Software as a Product (SaaP) to refer to the traditional software delivery model. We noticed that SaaP has been used in the same manner by some authors [3] [4] [5] [6] [7] although it is not very popular.

SaaP solutions require a user to purchase a license to use software that the user will then have to host. Examples of SaaP solutions include Microsoft Office and other on-premises software packages.

An SaaP solution does not require the Internet to run, but it needs the user to perform installation, configuration, and updating. These opreations could involve time cost and money cost.

SaaS has many advantages over SaaP, such as centrally-stored data, free maintenance, and free updates. From a user-cost point of view, SaaS is also very attractive. Analysis [3] shows that SaaS has a better economic model, which is lower in cost to the user and fundamentally inclined to improve for the user with each new release.

While SaaS is superior to SaaP in many aspects, it has one drawback: SaaS heavily relies on an Internet connection. Without an Internet connection, a user cannot do processing and cannot access existing data. For example, a store owner uses Salesforce [8] to manage store information. While systems are working smoothly, a local power failure makes the Internet connection unavailable. The owner wants or needs to use the laptop to find some important account and contact information stored in Salesforce, which might be vital in such a situation. Unfortunately, the owner cannot access the information because an Internet connection is unavailable. SaaP's major advantage over SaaS is that it does not rely on an Internet connection.

If the delivery method does not fit a user's needs, the user may reject the software. For example, a user who runs a store in an area where an Internet connection is not available or is unstable may have to say no to Salesforce and try to find an SaaP solution. On the other hand, many people stop using desktop email clients (SaaP) and turn to Web-based email services, such as Hotmail or Gmail (SaaS) because the features of SaaS are attractive to them.

While popular, many users are not completely satisfied with the two traditional software delivery models, SaaS and SaaP. Is it possible to combine the advantages of SaaS and SaaP so that a better software delivery model can be offered? In this paper, we explore the possibility of integrating SaaS and SaaP.

An immediate approach to the problems discussed above is to offer an SaaS solution and an SaaP solution at the same time. When both an SaaS solution and an SaaP solution are offered, users have more choices, but they still face limitations: In the SaaP solution, the data is saved in the local computer; in the SaaS solution, the data is saved in the central server; if a user wants to use both the SaaS solution and the SaaP solution, he/she has to maintain data consistency between the two databases by himself/herself. It is not easy for a user to keep his/her data being consistent in two databases. The user may use the SaaS solution for one set of data and use the SaaP solution for another set of data, but this arrangement may not satisfy all users.

If, for data consistency reasons, a user uses only one of these two offerings, SaaS or SaaP, the advantages of the other offering cannot be obtained by the user. Is it possible to link the two offerings in a way that users can take advantage of both of them? We propose to integrate SaaS and SaaP using the Web in Dew (WiD) category of dew computing [9].

2. Dew Computing, WiD, and Cloud-dew Architecture

Dew computing [9] [10] [11] is an emerging new research area; WiD [9] is a category of dew computing; cloud-dew architecture [11] provides a detailed framework to implement WiD. In this section, we introduce them briefly.

2.1. Dew Computing

Dew computing is defined in [9] as the following: "Dew computing is an on-premises computer software-hardware organization paradigm in the cloud computing environment where the on-premises computer provides functionality that is independent of cloud services and is also collaborative with cloud services. The goal of dew computing is to fully realize the potentials of on-premises computers and cloud services."

The term on-premises computers means any kind of computers as long as they are not part of cloud services. Thus, on-premises computers are almost the same with local computers. Roughly speaking, dew computing is a way of software design or a special kind of software for local computers. This kind of software has two key features: independence and collaboration.

Independence means the software must be able to work when no Internet connection is available. Collaboration means the software on the local computer must be able to automatically send/receive some information to/from the cloud when an Internet connection is available.

The word *dew* was associated with computing in an open access paper [11] available online in January 2015; what was proposed in this paper was Web-based and it is equivalent to the dew computing category WiD [9]. Later, a broader definition of dew computing was proposed [10]. Further definitions and descriptions about dew computing were reflected in [9] [12] [13]. Other major works in the dew computing area include some papers related to cloud-dew architecture [14] [15] [16], a scalable distributed computing hierarchy including cloud computing, fog computing, and dew computing [12], and the relationships among cloud computing, fog computing, and dew computing [17].

Dew computing is a new concept, but some dew computing applications were developed many years before this concept was proposed. Details can be found in [9].

2.2. WiD

Dew computing has many categories [9]. Almost all dew computing categories are named in the format of *X in Dew*, where X is a kind of resource or service; the basic meaning of X in Dew is that X somehow has its existence in an on-premises computer and X exchanges information with cloud services. In this paper, we are specially interested in one category of dew computing: Web in Dew (WiD).

The literal meaning of WiD is that the World Wide Web is put into an on-premises computer. Because this is impossible, we may want a fraction of the Web be put into an on-premises computer.

To implement WiD, we need to use the cloud-dew architecture [11].

2.3. Cloud-dew Architecture

Cloud-dew architecture [11] [15] [16] is a new architecture that was derived from client-server architecture. Client-server architecture [18] is a simple and elegant architecture of a network through which many clients (computers, computer programs, or devices) request and receive service from a centralized server (another computer, computer program, or device). Client-server architecture is the foundation of the Internet and the World Wide Web.

Cloud-dew architecture is an extension of the client-server architecture. In this extension, servers are classified further into cloud servers and dew servers. Cloud servers are the servers in the client-server architecture, which provide SaaS or similar cloud services. The newly-introduced dew servers are Web servers that reside on users' local computers. The dew servers and their related databases have two functions: first, they provide the client with the same services as those the cloud server provides; second, they synchronize dew server databases with cloud server databases automatically.

Figure 1 shows the core idea of cloud-dew architecture, which implements WiD.

2.4. Dew Computing Application Example

Let us consider the example we mentioned in Section 1. An owner of a small tourism store (the user) runs his business in a beach area where an Internet connection is not available or not stable, but every day after business hours, he goes home where an Internet connection is always available. He wants to use Salesforce for his business management in his laptop computer, but cannot because Salesforce relies on an Internet connection. The following discussions are based on the assumption that Salesforce has adopted the dew computing paradigm and created a WiD application using the cloud-dew architecture.

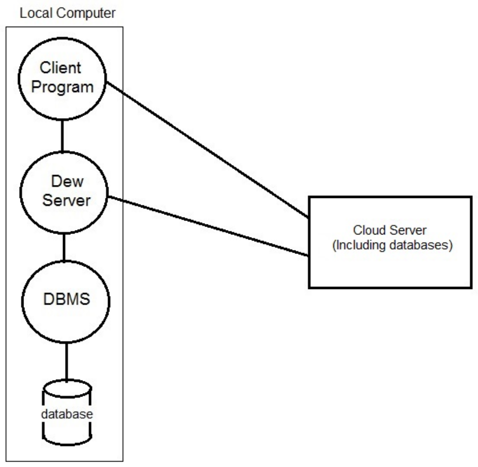

Figure 1. Cloud-dew architecture for WiD

The user can download and install two packages. The first package is a Web server, provided by Salesforce or another party. After the package is installed and started, a Web server runs on the user's laptop. This Web server serves only one user, and it is called a dew server. The second package contains all the code and other necessary resources for the dew server to host a *dewsite*, which is a mirror and simplified version of the Website http://www.salesforce.com. The second package is derived from the Website and would be provided to the user by Salesforce.

A dewsite is similar to the Website, but with the following differences:

- The dewsite does not need to deal with a heavy global load, so it will be much simpler than the Website;
- The dewsite will not include the proprietorial script that the vendor does not want to release. Instead, publicly-known technology will be used to implement similar functionality;
- The content of a dewsite database is limited;
- A new function, which will automatically synchronize local database with Website database, will be added to the dewsite.

A local Web server can be accessed using the URL http://localhost, but there is a better way to access it. With the help of the local domain name system [11] [16], the user is able to access this dewsite from a Web browser using the URL http://mmm.salesforce.com where *mmm* is used to indicate that this is a dewsite. Because the dew server is inside the local laptop, the dewsite is always available, with or without an Internet connection. The user can process all his business using this dewsite at the store location. When he returns to where an Internet connection is available, the dewsite will synchronize data with the Website http://www.salesforce.com automatically.

Now we can see that this application satisfies the independence and collaboration features. Dewsite http://mmm.salesforce.com provides independent Salesforce service; it also collaborates with Website http://www.salesforce.com to reach synchronization. This WiD application makes Web-surfing without an internet connection possible. When there is no internet connection, the user can still access the dewsite built in the laptop computer.

Readers may wonder why two packages instead of one package are needed. The reason is that the first package belongs to the infrastructure of dew computing and the second package is vendor-specific. This arrangement makes it easier for more WiD applications to be installed in the local computer.

3. The Integration of SaaS and SaaP

In the above example, the Website http://www.salesforce.com is an SaaS solution; the dewsite http://mmm.salesforce.com is an SaaP solution. The same software is provided to users in both SaaS and SaaP simultaneously. If an Internet connection is available, both the Website and the dewsite can be used; if there is no Internet connection, the Website is not available, however, the dewsite is still available. Any updates or additions to the data will be synchronized automatically with the Website when it is possible. In this manner, SaaS and SaaP are integrated and the software is delivered to users in a new way.

Generally speaking, SaaS and SaaP can be seamlessly integrated into a new software delivery model using the WiD category of dew computing; cloud-dew architecture provides the implementation framework for this new delivery model.

For this new software delivery model, we do not want to introduce another terminology. We may simply use WiD to denote this model. Thus, we may say that WiD is not only a category of dew computing, but also a new software delivery model. Section 3.1. to Section 3.3 will describe the attractive features of the WiD software delivery model.

3.1. Flexible Operation Mode

As discussed earlier, software simply delivered in both SaaS and SaaP without dew computing will not help users much because a user has to pick one and continue to use it due to data consistency reasons. The user cannot benefit from the advantages of the other delivery method. However, if the software is delivered through WiD, both SaaS and SaaP components are available. The user is able to move freely between operating on the SaaS component or operating on the SaaP component. This enables the user to take maximum advantage of both SaaS and SaaP.

For example, a user may register with Salesforce service and enjoy all the SaaS features. When an Internet connection is not available, the user can immediately switch to Salesforce dewsite http://mmm.salesforce.com, thereby accessing all data in the local database because of the automatic synchronization. During the period when there is no Internet connection, the user may update data or enter new data in the dewsite. When an Internet connection is available again, such data will be synchronized to the Website http://www.salesforce.com automatically.

Generally speaking, a user may choose to use SaaS or SaaP according to preference or situation. When the user is using his/her own computer with the dew server and the dewsite installed, both SaaS and SaaP can be used; when the user is using another computer, SaaS is the only option. When there is an Internet connection, both SaaS and SaaP can be used; when there is no Internet connection, SaaP is the only option.

3.2. Unified User Interface

By design, the interface of an SaaS solution must be in the form of Web pages. Previously, the interface of an SaaP solution had to be in the form of a desktop application. It is suggested that an SaaP solution can be built in the form of a local Website. With both applications now being in the form of Websites, the SaaS solution and the SaaP solution of the same software can be built with identical or near-identical interfaces. Such unified interfaces makes learning and training much easier.

3.3. Flexible Cost Model

Some authors have suggested that SaaS provides a better cost model than SaaP [3]. Regardless of whether this is true or not, WiD provides more flexibility in choosing cost models. For example, suppose that a user mainly uses particular software in SaaP (dewsite) fashion, it is natural for the user to purchase the software; the purchase can be recorded in the Website, and dewsite downloads for existing purchases will be granted whenever requested. If the same user prefers and the vendor agrees, the user can also pay the cost on subscription basis. In the Salesforce example, its cost model is subscription-based; after adopting the the dew computing paradigm, Salesforce can continue to charge fees on subscription basis, and the dewsite can be provided to members to expand the member base. A user's membership can be verified whenever the user contacts the Website, such as when a dewsite is downloaded or when synchronization is requested.

4. The Applications and Impacts of the New Software Delivery Model

By integrating SaaS and SaaP, a new software delivery model is constructed. This new software delivery model WiD will realize and maximize the advantages of SaaS and SaaP, expand software user base, and provide more satisfaction to users. The new software delivery model will have the following applications and impacts:

- WiD can be used to expand the user base of current software
  Similar to the Salesforce example discussed in Section 2 and Section 3, software and Websites may use WiD to expand their user base. Many software categories, such as word processing, social media, and business management, may find the benefits of WiD model. One of the challenges is that revisions to the existing system are necessary.
- WiD can be used as a design pattern for new software
  Newly-designed software can easily take advantage of WiD. The most significant point is that such software is considered as a distributed system using distributed databases from the beginning of its design.
- WiD will increase the demands to user computer hardware
  One side effect of cloud computing is that the demand to computer hardware does not increase as fast as before. If all of a user's data and software are in the cloud, a simple computer, which enables the user to connect to the Internet, is enough; many computing resources of the user's computer(s) become unnecessary. If WiD or other categories of dew computing is widely adopted, many dewsites will be available, and more computing resources will be needed to host these dewsites.

5. Conclusions

SaaP and SaaS are two major software delivery models. In the SaaP model, a user purchases software and manages it with actions such as installation, configuration, and updating; the advantage of SaaP is that software is installed locally so that it is always available with or without an Internet connection. In the SaaS model, a user does not need to manage the software; moreover, the software can be accessed anywhere as long as an Internet connection is available. However, the drawback is that if there is no Internet connection, the software and user data will not be available.

Integrating SaaP and SaaS is a natural progression in finding a better software delivery model. The WiD category of dew computing can be used to integrate SaaS and SaaP, and we use WiD to denote this new model. Thus, WiD is not only a category of dew computing, but also a new software delivery model which integrates SaaS and SaaP. WiD software delivery model has some great features, such as flexible operation mode, unified user interface, and flexible cost model.

References

[1] Software as a service (saas). Gartner IT Glossary. [Online]. Available: http://www.gartner.com/it-glossary/software-as-a-service-saas/

[2] B. P. Rimal, E. Choi, and I. Lumb, "A taxonomy and survey of cloud computing systems," in *INC, IMS and IDC, 2009. NCM '09. Fifth International Joint Conference on*, Aug 2009, pp. 44–51.

[3] S. Sehlhorst. (2008, Nov.) The economics of software as a service (saas) vs. software as a product. Pragmatic Marketing. [Online]. Available: http://pragmaticmarketing.com/resources/the-economics-of-software-as-a-service-saas-vs-software-as-a-product

[4] R. Rajala, M. Rossi, and V. K. Tuunainen, "A framework for analyzing software business models," in *Proceedings of the 11th European Conference on Information Systems (ECIS)*, Naples, Italy, Jun. 2003, pp. 1614–1627.

[5] K. Popp, "Software industry business models," *IEEE Software*, vol. 28, no. 4, pp. 26–30, July 2011.

[6] T. Heart, N. S. Tsur, and N. Pliskin, *Global Sourcing of Information Technology and Business Processes: 4th Global Sourcing Workshop 2010, Zermatt, Switzerland, March 22-25, 2010, Revised Selected Papers*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, ch. Software-as-a-Service Vendors: Are They Ready to Successfully Deliver?, pp. 151–184.

[7] S. A. Mokhtar, S. H. S. Ali, A. Al-Sharafi, and A. Aborujilah, "Cloud computing in academic institutions," in *Proceedings of the 7th International Conference on Ubiquitous Information Management and Communication*, ser. ICUIMC '13. New York, NY, USA: ACM, 2013, pp. 2:1–2:7. [Online]. Available: http://doi.acm.org/10.1145/2448556.2448558

[8] Salesforce. salesforce.com inc. [Online]. Available: http://www.salesforce.com

[9] Y. Wang, "Definition and categorization of dew computing," *Open Journal of Cloud Computing (OJCC)*, vol. 3, no. 1, pp. 1–7, 2016.

[10] Y. Wang, (2015, Nov.) The initial definition of dew computing. Dew Computing Research. [Online]. Available: http://www.dewcomputing.org/index.php/2015/11/10/the-initial-definition-of-dew-computing/

[11] Y. Wang, "Cloud-dew architecture," *International Journal of Cloud Computing*, vol. 4, no. 3, pp. 199–210, 2015.

[12] K. Skala, D. Davidovic, E. Afgan, I. Sovic, and Z. Sojat, "Scalable distributed computing hierarchy: Cloud, fog and dew computing," *Open Journal of Cloud Computing*, vol. 2, no. 1, pp. 16–24, 2015.

[13] S. Ristov, K. Cvetkov, and M. Gusev, "Implementation of a horizontal scalable balancer for dew computing services," *Scalable Computing: Practice and Experience*, vol. 17, no. 2, pp. 79–90, 2016.

[14] D. Bradley. (2015, Sep.) Dew helps ground cloud services. Science Spot. [Online]. Available: http://sciencespot.co.uk/dew-helps-ground-cloud-services.html

[15] Y. Wang and Y. Pan, "Cloud-dew architecture: realizing the potential of distributed database systems in unreliable networks," in *Proc. The 21st International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA15)*, Las Vegas, USA, Jul. 2015, pp. 85–89.

[16] Z. Kang, "A new method to implement ldns in the cloud-dew architecture," University of Prince Edward Island, CSIT Research Report CS-21, Nov. 2015.

[17] Y. Wang. (2015, Nov.) The relationships among cloud computing, fog computing, and dew computing. Dew Computing Research. [Online]. Available: http://www.dewcomputing.org/index.php/2015/11/12/the-relationships-among-cloud-computing-fog-computing-and-dew-computing/

[18] client/server. Gartner IT Glossary. [Online]. Available: http://www.gartner.com/it-glossary/clientserver/