More Tales of Clouds: Software Engineering Research Issues from the Cloud Application Perspective*

Lijun Mei
The University of Hong Kong
Pokfulam, Hong Kong
ljmei@cs.hku.hk

Zhenyu Zhang
The University of Hong Kong
Pokfulam, Hong Kong
zyzhang@cs.hku.hk

W.K. Chan†
City University of Hong Kong
Tat Chee Avenue, Hong Kong
wkchan@cs.cityu.edu.hk

***Abstract*—Cloud computing is an emerging computing paradigm. It aims to share data, calculations, and services transparently among users of a massive grid. Although the industry has started selling cloud-computing products, the software engineering infrastructure and application model are still unclear. In this paper, we compare cloud computing with service-oriented computing and pervasive computing. Both the industry and research community have actively examined these three computing paradigms. In this paper, we draw a qualitative comparison among their application characteristics based on the classic model of computer architecture from the cloud application perspective. We contrast the difference and pinpoint areas that researchers may examine in the future.**

Keywords—cloud computing; software engineering; applications

I. INTRODUCTION

Cloud computing [24][27] shares transparently computation, data and resources over a scalable network of nodes. Examples of such nodes include data centers and web services. *Computing cloud* or simply *cloud* is the usual name to refer to such a network of nodes. We call an application building on such a platform as a *cloud application*.

Industrial leaders such as Microsoft [21], Google [2], and IBM [14] are engaged in the paradigm. There are industrial conferences such as Cloud Computing Conference & Expo [6]. Gartner *Research* further comments that "*cloud computing will be as influential as e-business*" [11]. We wonder what the viewpoints of software engineering and programming (SE&P) research community on cloud application are.

Despite the vast interest on cloud computing in industry, we only find little public academic literature in this area. In particular, Mei et al. [19] discuss the issues from the general paradigm viewpoint. We recapture their observations as follows:

Incubation duration. Service-oriented computing waits for a long time to receive supports from industry, such as IBM [15], Microsoft [20], and BEA [24]. Similarly, it has been many years since the early formalization effort [5] of cloud computing.

Mature supportive technology. Service-oriented architecture (SOA) has become popular since the success of web services. Similarly, many techniques for cloud computing have been mature [8][9]. Companies have begun to ship cloud computing machines [8][14]. News reveals that Microsoft may file a patent on pay-by-use for its software [30].

Practice ahead research. In both service-oriented computing and cloud computing, SE&P research lags behind industrial adoptions. For instance, *COSCON* has successfully adopted SOA to improve the response time to customers in 2004 [3].

The above trends lead us to study application development challenges in cloud computing in two areas, namely the early vision of the computing paradigm, and the supportive technology. We follow the presentation style in [19] to present our findings. Specifically, we use the classic computer architecture model [13] to provide a qualitative comparison framework to compare cloud application with pervasive application and service-oriented application. We assume that readers carry the background knowledge of service-oriented computing and pervasive computing.

We choose to compare with pervasive computing because pervasive computing has a high-contrast development direction in the above three trends: Pervasive computing needs much longer incubation period (e.g., the seminal paper [29] of Mark Weiser in 1991), supportive technologies are still immature, and applications have not been widely available in industry. We view that such differences among the three paradigms provide a basis in our comparison on the similarity and differences of their applications needs on the foundations of their infrastructure.

The main contribution of the paper is twofold: (i) We provide a qualitative comparison on application development support for the three paradigms. (ii) The comparison reveals research issues for cloud application research and development, particular in the SE&P area.

The rest of the paper is organized as follows. Section II presents the preliminaries of cloud computing and two other computing paradigms. Section III presents the framework and our findings. We review related work in Section IV, followed by a conclusion in Section V.

* This research is supported in part by GRF grants of the Research Grants Council of Hong Kong (project nos. 111107, 123207, 717308, and 717506).

† Correspondence Author

II. PRELIMINARIES

This section reviews the preliminaries of cloud computing, services computing, and pervasive computing.

A. Preliminaries of Cloud Computing

Cloud application is a kind of applications similar to existing web applications such as Gmail and Yahoo! Mail, yet it aims to support to access data or software by encapsulating when, what, where, how, by whom provide the computational and storage resources. To ease development, *platform* (or *infrastructure*) such as Google App Engine [12] is being proposed.

For a cloud, the scalability (both horizontal scalability and vertical scalability [19]) should ideally be transparent to application users. For instance, users may store their data in the cloud without the need to know the location in the cloud to keep their data or differentiate technologies (or parameters) used by the cloud to accesses their data.

B. Preliminaries of Service-Oriented Computing

From the application perspective, service-oriented computing (or services computing) models, develops, executes, and maintains services. A *service* is a function with well-defined, machine-understandable interfaces and uses standardized machine-readable messages to communicate with external entities. A *service composition* is a set of services that they communicate with one another in a "program execution" so that these services can output results to their users. In a service composition, the role *service consumer* (e.g., [17]) refers to a service that sends a message (known as a *service request*) to the peer services, the role of a service receiving such a message is called a *service provider*. If a service provider receives a service request, it will execute its function, and then replies its outputs as messages to the target service consumer. Such a reply message is known as a *service response*.

C. Preliminaries of Pervasive Computing

Pervasive computing (also known as ubiquitous computing) [25] is also a computing paradigm that application adapts to users, rather than the other way round. From the application viewpoint, pervasive software is embedded in a constantly changing computing environment of users. Based on the environment attributes, the applications adapt their behavior accordingly. A well-developed environment might enable users to use pervasive software everywhere without extra effort.

To serve a user and react to a user action, applications use environmental attributes, known as contexts, extensively. Sensors can capture these contexts. To allow ubiquitous support to users, such sensors are placed around users to capture different information, such as user locations and user activity.

III. PARADIGMS COMPARISON ON THE CLASSICAL COMPUTER ARCHITECTURE MODEL

A. Backgrounds

In the subsequent sub-sections, we are going to present a qualitative comparison among the cloud, services, and pervasive computing applications. Existing work [23] had compared cloud computing with grid computing, and compared cloud computing from the paradigm viewpoint [19]. Viewing the same topic from multiple perspectives is an essential step toward comprehensive study of cloud computing.

In this paper, we compare applications of the three paradigms using the classic computer architecture model [13], which has three features: input-output (I/O), storage, and calculation. From the software application perspective, the input-output feature captures the notion that a computation requires inputs and outputs. Conventional inputs to software include user inputs via user interface, system inputs via operating systems, data from file systems

TABEL 1. OVERALL COMPARISON

	Feature	**Key Conceptual Characteristics**	**Similar to Cloud Computing?**
Cloud	I/O	Client requests and server responses.	
	Storage	Stored in the clouds collectively.	
	Calculation	Performed by individual cloud applications.	
Service	I/O	Service requests and service responses.	✓
	Storage	Application code and data kept at specific servers.	
	Calculation	Performed by individual service or service compositions.	✓
Pervasive	I/O	Applications generate and use contexts.	
	Storage	Contexts stored in the tuple space of an application.	✓
	Calculation	Performed by individual applications or applications sharing contexts.	✓

TABEL 2. COMPARISONS IN THE I/O FEATURE.

	I/O Attribute	Key Characteristics	Related Research Issues
Cloud	Interface	Unclear.	How do applications specify the standardized application interface for cloud connection? Do cloud applications reuse the semi-structural data type notion as for services? What is the access mode for cloud applications?
	Data Type	Unclear.	
	Access mode	Unclear.	
Service	Interface	Standardized service interface (e.g., WSDL).	
	Data Type	Semi-structural and in the XML format (e.g., XML Schema).	
	Access mode	Service request and responses.	
Pervasive	Interface	Contexts as interface definition.	
	Data Type	Diversified context types (e.g., XML, WAP, GPRS, and Bluetooth).	
	Access mode	Through context communications.	

TABEL 3. COMPARISONS IN THE STORAGE FEATURE.

	Storage Attributes	Key Characteristics	Related Research Issues
Cloud	Location	Transparent.	How can developers track the control flow and data flow in applications? How does an application manipulate data structures (such as entities in context-like repository)?
	Size	Transparent.	
	Repository	Unclear.	
Service	Location	Storage encapsulated within individual services and at specific servers. Explicit boundaries among individual services in a service composition.	
	Size	Limited within individual services.	
	Repository	No concept of repository to keep messages among services, even within a service composition.	
Pervasive	Location	In the surrounding environment of an application.	
	Size	The surrounding computing environment of an application.	
	Repository	Use tuple space to keep contexts.	

TABEL 4. COMPARISONS IN THE CALCULATION FEATURE.

	Calculation Attributes	Key Characteristics	Related Research Issues
Cloud	Concurrency	Unclear.	What is the first-class entity in cloud application? What is an appropriate concurrency model for cloud applications? How can developers collect the execution states of cloud applications?
	Execution	Abstract execution trace may be known.	
	Granularity	Unclear.	
Service	Concurrency	Choreography.	
	Execution	Service execution trace can be known.	
	Granularity	A service is an execution unit.	
Pervasive	Concurrency	Standard concurrency model.	
	Execution	Program execution trace can be known.	
	Granularity	Any computing entity is an execution unit.	

or the other software entities. Similarly, conventional software outputs include web pages, or data sent to the other software entities. The storage feature represents the volatile and persistent storage (e.g., memory or disk) so that the computer may use them to keep data such as the codes and

artifacts of an application. The calculation feature represents the computation capability of the computer.

Having introduced the basics of the dimensions in the classical computer architecture, in the rest of this section, we summarize our findings.

B. Overall Key Characteristics of Applications in the Three Paradigms

Table 1 shows a summary of key characteristics of the three features in each paradigm from the application viewpoint. We observe a number of similarities among characteristics from Table 1.

- The I/O feature of cloud applications resembles that of services applications.
- The storage feature of cloud applications resembles to that of pervasive applications.
- The calculation features of the applications in the three computing paradigms are similar.

This finding is in line with the overall finding presented in [19]. To study cloud applications, we thus further use the above-identified similarities to examine these characteristics of service-oriented computing and pervasive computing. Although the three paradigms are similar at this level, they show differences in the details in terms of application support, as we will present below.

Tables 2, 3, and 4 show a series of comparisons of the key characteristics in the I/O, storage, and calculation features, respectively. In the following three sub-sections, we discuss them in turn.

C. Comparison on the I/O feature

Table 2 summarizes the comparison on the I/O features. We provide three sub-features under the I/O feature to compare these computing paradigms. To quantify an I/O for an application, we consider the input/output parameter quantification, which includes both the data type and the interface. Moreover, to connect to such applications, developers want to know the access mode.

As we have mentioned in Section III(B), the I/O feature of cloud applications resembles to that of service applications. In this connection, should applications of cloud computing follow services to use standardized interfaces and messages in the semi-structural form? On the other hand, pervasive computing shows that using implicit invocations can be a means to access contexts. Thus, should cloud applications use implicit invocations of components (via contexts) or semi-structural data?

D. Comparison on the Storage feature

We give the comparisons on the storage feature in Table 3. Three subfeatures are listed for each computing paradigms: location, size and repository. We choose to compare these three subfeatures because applications need to know the size and format of the repository so that they can use the repository efficiently and effectively. We include "location" in the storage feature comparisons because location-transparency is a key concept in cloud computing.

Recall that Table 1 has outlined pervasive applications similar to cloud applications. We thus proceed to contrast service-oriented applications against pervasive applications to identify research directions for cloud applications. In particular, pervasive applications manipulate contexts in a tuple space surrounding a user. Thus, how a cloud application manipulates data structures (such as entities in context-like repository) is unclear. On the other hand, service compositions tend to encapsulate the storage within individual services. As such, control flow and data flow of services can be tracked within a service. To support development of cloud applications, one might wonder how to track the control flow and data flow in such cloud applications.

E. Comparison on the Calculation feature

We continue to compare cloud applications with the applications in the other two computing paradigms to gain more insights in the characteristics of calculation feature. Table 4 summarizes the results.

In service-oriented computing, service is a first-class entity; in pervasive computing, context is a first-class entity. In either paradigm, applications can use (or access) these entities *in their applications*. We thus ask the following question: What are the first-class entities in cloud applications?

Even though such kinds of first-class entity have been clarified, researchers would like to model or collect the execution states of cloud applications. However, in cloud computing, such an execution can be globally distributed. For example, the Cloud Computing Test Bed [16] promoted by HP, Intel and Yahoo! provides a globally distributed, Internet-scale execution environment, data center management and hardware issues associated with cloud computing. What is a state in cloud application? By a similar token, it is unclear what an appropriate execution model for cloud applications is.

IV. Related Work

This section reviews the literatures that are related to our work.

First, we review grid computing in general. The paradigm of grid computing is close to that of cloud computing [28]. Foster and Kesselman [10] present their understanding of grid computing. They [10] show how grids can solve research problems such as diagnostic problems and Aero-engine DP problems. Other researches on grid computing, such as [9], focus on the computing entity organization and computing task distribution. It would be difficult for an ordinary user to make use of such grid services. On the contrary, cloud computing highlights user experience in cloud services and encourages any users to use cloud services as if they were using their own computing laptops or PCs.

Second, we review context-aware approaches in general. Context-aware approaches are important in providing adaptive behaviors to software applications. Tse et al. [26] propose to use isotropic properties of contexts as metamorphic relations for testing context-sensitive software. Mokhtar et al. [22] explain the problem of composition in the environment of pervasive computing. Besides, Anhalt et al. [1] present an approach to representing context awareness in programs.

For cloud computing, large-scale computing entities will be placed on various host machines with different environmental contexts. The environmental contexts may heavily affect the performance of clouds. To guarantee the quality of cloud services, therefore, it is critical to develop techniques to address the environmental effect.

In service-oriented computing, Castagna et al. [4] develop a process-centric theory to compose and replace services. Mei et al. propose to address the service selection problem by link analysis [17], and develop a technique to test BPEL web services [18].

The interface of cloud computing has been considered to be an important issue to study cloud computing. For instance, in the unified cloud interface (UCI) [7], a cloud broker is used as a common interface for the interaction with remote computing facilities such as platforms, networks, data, and services. It is composed of a specification and a schema. The broker schema provides the cloud model descriptions (defined by a common set of cloud definitions). The specification defines how cloud model can be integrated with other management models at the specification-level. UCI aims to provide a decentralized and extensible framework over secure asynchronous communication among brokers.

V. Concluding Remarks

In this paper, we have drawn up a qualitative comparison framework to compare cloud applications with pervasive applications and service-oriented applications. This framework positions on the classic model of computer architecture, which includes three features: input-output, storage and calculation.

We view that a cloud application demands an underlying theory and framework to quantify the key first-class entities, and explore how to use such first-class entities and communicate them with the other applications (or clouds). In this connection, these applications may use standardized protocol, semi-structural data schema, and data descriptor to quantity such first-class entities, and transform them as "virtualized entities". For such virtualized entities, tracking their control flow and data flow on certain user-transparent concurrent model seems demanding more research studies. Moreover, because the full execution states of a cloud application is only partially accessible to the cloud application, the way to model and monitor program states and state transitions may be interesting to explore.

References

[1] J. Anhalt, A. Smailagic, D. P. Siewiorek, F. Gemperle, D. Salber, S. Weber, J. Beck, and J. Jennings. Toward context-aware computing: experiences and lessons. *IEEE Intelligent Systems*, 16 (3): 38–46, 2001.

[2] S. Baker. Google and the wisdom of clouds. Available at `http://www.businessweek.com/magazine/content/07_52/b4064048925836.htm`. (Last accessed on April 30, 2009)

[3] M. Barnes. COSCON improves business responsiveness via service-oriented architecture. Gartner Research, 2006. Available at `http://www-128.ibm.com/developerworks/blogs/resources/SOA_Off_the_Record/coscon_improves_business_res_139656.pdf`. (Last accessed on April 30, 2009)

[4] G. Castagna, N. Gesbert, and L. Padovani. A theory of contacts for web services. In *Proceedings of Principles of Programming Languages (POPL 2008)*, 261–272. 2008.

[5] R. Chellappa. Cloud computing: emerging paradigm for computing. In *INFORMS 1997*. Dallas, TX, 1997.

[6] Cloud Computing Conference & Expo. Available at `http://cloudcomputingexpo.com/`. (Last accessed on April 30, 2009).

[7] R. Cohen. Unified Cloud Interface (UCI). Available at `http://groups.google.ca/group/cloudstorage`. (Last accessed on April 30, 2009)

[8] Dell cloud computing solutions. Available at `http://www.dell.com/cloudcomputing`. (Last accessed on April 30, 2009)

[9] I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the grid: enabling scalable virtual organization. *International Journal of High Performance Computing Applications*, 15 (3): 200–222, 2001.

[10] I. Foster and C. Kesselman. *The Grid: Blueprint for a New Computing Infrastructure*. Elsevier, Amsterdam, 2004.

[11] Gartner. Gartner says cloud computing will be as influential as e-business. 2008 Press Release. Available at `http://www.gartner.com/it/page.jsp?id=707508`. (Last accessed on April 30, 2009).

[12] Google App Engine. Available at `http://code.google.com/appengine/`. (Last accessed on April 30, 2009)

[13] J. P. Hayes. *Computer Architecture and Organization*. McGraw-Hill, New York, 1998.

[14] IBM. Big blue goes for the big win. Available at `http://www.businessweek.com/magazine/content/08_10/b4074063309405.htm`. (Last accessed on April 30, 2009)

[15] IBM. Cloud computing: gain access to your applications from anywhere, at any time. Available at `http://www.ibm.com/ibm/cloud/`. (Last accessed on April 30, 2009).

[16] Infocomm Development Authority of Singapore and University of Illinois at Urbana-Champaign and Karlsruhe Institute of Technology. HP, Intel and Yahoo! Create Global Cloud Computing Research Test Bed. Available at `http://www.businesswire.com/news/google/20080729005585/en`. (Last accessed on April 30, 2009.)

[17] L. Mei, W. K. Chan, and T. H. Tse. An adaptive service selection approach to service composition. In *Proceedings of*

the IEEE International Conference on Web Services (*ICWS*), pages 70–77. 2008.

[18] L. Mei, W. K. Chan, and T. H. Tse. Data flow testing of service-oriented workflow applications. In *Proceedings of the 30th International Conference on Software Engineering* (*ICSE 2008*), pages 371–380. 2008.

[19] L. Mei, W.K. Chan, and T.H. Tse. A tale of clouds: paradigm comparison and some thoughts on research issues. In *Proceedings of 2008 IEEE Asia-Pacific Services Computing Conference (APSCC 2008)*, pages 464–469. 2008.

[20] Microsoft Corporation. Service orientation and its role in your connected systems strategy. Available at `http://msdn.microsoft.com/en-us/library/ms954826.aspx`. (Last accessed on April 30, 2009)

[21] Microsoft Corporation. Software via the Internet: Microsoft in 'cloud' computing. Available at `http://www.nytimes.com/2007/09/03/technology/03cloud.html`. (Last accessed on April 30, 2009)

[22] S. B. Mokhtar, D. Fournier, N. Georgantas, and V. Issarny. Context-aware service composition in pervasive computing environments. In *Rapid Integration of Software Engineering Techniques*, volume 3943 of Lecture Notes in Computer Science, pages 129–144. 2006.

[23] P.S. Narayanan. From grid computing to cloud computing: the IBM approach. Garuda Partner Meet, Bangalore, India, March 4, 2008.

[24] Y.V. Natis. Genesis Marks BEA's Strategic Turn to 'Cloud Computing'. Gartner research, ID number G00154318. 2007.

[25] M. Satyanarayanan. Pervasive computing: vision and challenges. *IEEE Personal Communications*, 8 (4): 10–17, 2001.

[26] T.H. Tse, S.S. Yau, W.K. Chan, H. Lu, and T.Y. Chen. Testing context-sensitive middleware-based software applications. In *Proceedings of 28th International Computer Software and Applications Conference* (*COMPSAC 2004*), pages 458–466. 2004.

[27] M.A. Vouk. Cloud Computing – Issues, Research and Implementations. In *Proceedings of the 30th International Conference on Information Technology Interfaces* (*ITI 2008*), pages 31–40. 2008.

[28] P. Wallis. Cloud computing: is the cloud there yet? — a brief history. Available at `http://soa.sys-con.com/read/581838.htm`. (Last accessed on April 30, 2009)

[29] M. Wesier. The computer for the 21 century. Special Issue on Communications, Computers, and Networks, *Scientific American*, 1991.

[30] ZDNet. Microsoft files pay-per-use PC patent. Available at `http://news.zdnet.com/2424-9595_22-256995.html`. (Last accessed on April 30, 2009).