CLOUDQUAL: A Quality Model for Cloud Services

Xianrong Zheng, *Student Member*, *IEEE*, Patrick Martin, Kathryn Brohman, and Li Da Xu, *Senior Member*, *IEEE*

***Abstract*—Cloud computing is an important component of the backbone of the Internet of Things (IoT). Clouds will be required to support large numbers of interactions with varying quality requirements. Service quality will therefore be an important differentiator among cloud providers. In order to distinguish themselves from their competitors, cloud providers should offer superior services that meet customers' expectations. A quality model can be used to represent, measure, and compare the quality of the providers, such that a mutual understanding can be established among cloud stakeholders. In this paper, we take a service perspective and initiate a quality model named CLOUDQUAL for cloud services. It is a model with quality dimensions and metrics that targets general cloud services. CLOUDQUAL contains six quality dimensions, i.e., usability, availability, reliability, responsiveness, security, and elasticity, of which usability is subjective, whereas the others are objective. To demonstrate the effectiveness of CLOUDQUAL, we conduct empirical case studies on three storage clouds. Results show that CLOUDQUAL can evaluate their quality. To demonstrate its soundness, we validate CLOUDQUAL with standard criteria and show that it can differentiate service quality.**

***Index Terms*—Cloud computing, Internet of Things (IoT), quality model, validity criteria.**

I. Introduction

INTERNET OF THINGS (IoT) has emerged as the next revolutionary technology in the information industry [6], [12]. IoT allows objects like computers, sensors, mobile phones, etc. to communicate via the Internet. It has the potential to transform the current static Internet into a fully integrated future Internet. Cloud computing is an important component of the backbone of the IoT. Clouds will be required to support large numbers of interactions with varying quality requirements. Service quality will therefore be an important differentiator among cloud providers.

Manuscript received September 14, 2013; revised January 13, 2014; accepted January 28, 2014. Date of publication February 14, 2014; date of current version May 02, 2014. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC), in part by the Social Sciences and Humanities Research Council of Canada (SSHRC), in part by the National Natural Science Foundation of China (NSFC) under Grant 71132008, and in part by U.S. National Science Foundation (NSF) under Grant SES-1318470 and Grant 1044845. Paper no. TII-13-0635.

X. Zheng and P. Martin are with the School of Computing, Queen's University, Kingston, ON K7L 2N8, Canada (e-mail: xianrong@cs.queensu.ca; martin@cs.queensu.ca).

K. Brohman is with the School of Business, Queen's University, Kingston, ON K7L 3N6, Canada (e-mail: kbrohman@business.queensu.ca).

L. D. Xu is with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China; with Shanghai Jiao Tong University, Shanghai 200240, China; with the University of Science and Technology of China, Anhui 230026, China; and also with Old Dominion University, Norfolk, VA 23529 USA (e-mail: lxu@odu.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2014.2306329

Clouds are now a new battlefield—IT giants from Amazon to Google to IBM to Microsoft have entered the cloud market to acquire new customers and expand their businesses. Cloud services mean X as a Service (XaaS), where X can be hardware, software, and applications [1], [21]. In order to succeed in the competitive market, cloud providers should offer superior services that meet customers' expectations. Unlike traditional services, cloud services are delivered in Internet-based environments, with little or no direct human interaction. As a result, how to define and measure their quality becomes a new problem.

A quality model of cloud services specifies quality dimensions and metrics to detail and quantify service quality. It helps to create common knowledge of service quality, i.e., what it means and how to measure it, such that when a quality dimension like reliability is mentioned, it means exactly the same thing to two parties and the same metric is adopted to measure it. With a quality model, cloud consumers can confirm whether services are provided with the expected quality, and can eliminate potential misrepresentation. So, a quality model is able to protect cloud consumers' interests.

In fact, without a quality model, it is hard to establish trust between two parties, which is essential to conducting business in Internet-based environments. If cloud providers still make little or no commitment [3], as they do today, it inevitably harms their businesses. As cloud consumers care about quality of service (QoS), those who do not adopt a quality model and make no commitment in today's market may not survive. Indeed, without a quality model, cloud providers cannot assure themselves that their truly superior services are not repudiated. So, a quality model benefits cloud providers too and makes possible a large-scale adoption of cloud services.

The paper's main contributions are as follows.

1) A quality model for cloud services, called CLOUDQUAL, which specifies six quality dimensions and five quality metrics: It is a model with quality dimensions and metrics that targets general cloud services.
2) A case study involving three real-world storage clouds: Our experimental results show that CLOUDQUAL can evaluate their quality, which demonstrates its effectiveness.
3) A method to formally validate a quality model using standard criteria, namely, correlation, consistency, and discriminative power [9]: We show that CLOUDQUAL can differentiate service quality, which demonstrates its soundness.

The rest of the paper is structured as follows. Section II reviews related work. Section III presents CLOUDQUAL, a quality model that we propose for cloud services, which specifies six quality dimensions and five quality metrics. Section IV

describes our empirical case studies on three storage clouds. Section V validates CLOUDQUAL with three criteria, i.e., correlation, consistency, and discriminative power. Section VI summarizes the paper and mentions potential limitations.

II. Related Work

IoT allows connected objects to communicate via the Internet, whereas cloud computing promises unlimited resources delivered over the Internet [11], [12]. Gubbi *et al.* [6] present a cloud centric vision for worldwide implementation of IoT, where key enabling technologies and application domains are discussed. The vision comprises a flexible and open architecture that enables different players to interact in the IoT framework.

In studying service science, many ideas and techniques have been proposed [20], [22], [23], [27]–[29]. A service is regarded as an activity rather than a physical object, and so has four unique features, i.e., intangibility, heterogeneity, inseparability, and perishability [15], [16]. Unlike goods or products, which are tangible and have physical dimensions, services are "intangibles" whose output is viewed as an experience. So, it is hard to determine their quality.

In order to measure service quality, several quality models are proposed [14], [16]. SERVQUAL was developed in 1988 for measuring service quality in traditional services [14]. It enables service and retail businesses to evaluate consumer perceptions of service quality, and helps them to identify areas that need improvements. SERVQUAL includes five quality dimensions, i.e., tangibles, reliability, responsiveness, assurance, and empathy.

Networked software applications that perform business activities are termed electronic services or e-services for short [16]. From the marketing perspective, e-service quality means the extent to which the Internet facilitates efficient delivery of products and/or services. Swaid and Wigand [16] restructure the quality dimensions of SERVQUAL and propose a quality model for e-services. It mainly consists of six quality dimensions, i.e., website usability, information quality, reliability, responsiveness, assurance, and personalization.

QoS is important in discovering, selecting, and composing Web services [7], [18], grid services [17], and cloud services [8], [10], [19]. Wang *et al.* [24] argue that QoS is increasingly emphasized, modeled, and monitored in enterprise cloud services. Garg *et al.* [5] argue that with the growth of public cloud offerings, multiple services with the same function but different quality attributes are available, and that it is important to evaluate them, in an objective way, to find the most suitable one.

However, the above two models, whose quality dimensions are subjective, cannot be applied to cloud services without re-establishment. In fact, it would be better to define a quality model specifically for cloud services. Initial efforts in this direction are discussed below. Ferretti *et al.* [4] design a middleware that can respond effectively to QoS requirements of customer applications, which are hosted in an execution platform constructed out of cloud resources. As to quality dimensions, only availability and response time are considered, but important ones like reliability and security are not covered.

In order to help customers select cloud services that meet their needs and create healthy competition among cloud providers, Garg *et al.* [5] propose a framework called SMICloud to measure QoS for cloud services. SMICloud is based on Service Measurement Index (SMI). Although SMI formally specifies quality dimensions, it does not define any quality metrics. As SMICloud lacks sound justifications, important quality dimensions like security are left out. Finally, SMICloud targets Infrastructure as a Service (IaaS) in particular, not cloud services in general.

III. Cloud Quality Model

Cloud services are carried out in Internet-based environments. For this reason, they share few similarities with traditional services that are delivered in human-based environments. Instead, they share more similarities with e-services that are delivered in Internet-based environments too. Unlike traditional services, which are human powered services, cloud services are machine powered services, whose quality is not tightly linked to the performance of service employees, and so can be engineered. As such, cloud services require objective quality dimensions, with which cloud consumers can compare QoS delivered with QoS promised by cloud providers.

The model proposed by Swaid and Wigand [16] is an effective tool to evaluate e-service quality. It enables consumers to rate service experience for e-services. As its six quality dimensions are all subjective, it suffers the same drawback as SERVQUAL—it is unable to offer objective quality measurements. So, it cannot be applied to cloud services without reconstruction. As cloud services advocate computation, in a general meaning, as a service, objective quality dimensions are required to measure their quality. So, it is necessary to rebuild the model for cloud services. A quality model for cloud services should be *objective, computable*, and *verifiable*, so that cloud providers can gauge the QoS delivered, and cloud consumers can validate the QoS received.

A. Functional Versus Nonfunctional Properties

For cloud services, *functional properties* detail what is offered. For instance, Amazon Simple Storage Service (Amazon S3) provides storage services; Amazon Elastic Compute Cloud (Amazon EC2) offers compute services. In fact, if functional properties fail, cloud consumers' requirements cannot be fulfilled. So, it is not surprising that functional properties have received a great deal of attention. In contrast, *nonfunctional properties* detail how well a service is performed. For instance, Amazon S3 guarantees "a monthly uptime percentage of at least 99.9% during any monthly billing cycle" [30]. Here, an availability of at least 99.9% is promised, which is one of the important nonfunctional properties of cloud services.

Cloud vendors, however, do not yet give similar consideration to nonfunctional properties of their services. Of the Service Level Agreements (SLAs) typically specified with cloud services, most deal with availability and some consider reliability [3]. Indeed, nonfunctional properties matter because they determine service quality. For instance, if a network connection breaks down or performance becomes poor, it may affect availability. Also, if hardware failures or software faults occur, they may decrease reliability. Still, if attacks or intrusions happen, they may harm security. In short, if nonfunctional properties become

problematic, service experience can be poor, which negatively impacts a provider's reputation.

B. Cloud Quality Dimensions and Metrics

Inspired by SERVQUAL and the e-service quality model described earlier, we propose the following quality dimensions and metrics for cloud services.

1) Usability: Usability (USAB) describes how easy, efficient, and enjoyable the interface to a cloud service is to use, or assesses the ease of invocation if the functionality of a cloud service is exposed as Application Programming Interface (API).

For an end user who has no expertise in cloud services, a Graphical User Interface (GUI) serves better than an API. It should be noted that this is typical of most cloud consumers, and cloud providers cannot achieve business success without treating them as first-class citizens.

Still, a Web User Interface (WUI) is better than a GUI. End users have to install a client GUI to visualize their files stored in a storage cloud, whereas a WUI does not require extra effort from users.

In short, cloud interfaces should not cause too much cognitive pain to end users. Accurate and helpful information can aid users to interact with them. Since it is hard to give usability a quantitative description, it remains subjective.

2) Availability: Availability (AVAL) is the uptime percentage of cloud services during a time interval, which can be measured by

$$\alpha = \frac{t}{t_s} \tag{1}$$

where $0 \leq \alpha \leq 1$ represents availability; t and t_s denote the uptime and the total time of an operational period, respectively. The closer the value of α is to 1, the higher the availability.

As cloud services are delivered over the Internet, where network outages could happen, consumers value a highly available service. In other words, cloud services, ideally, should be interruption-free.

3) Reliability: Reliability (REL) is the assurance that cloud services are free from hardware failures, software faults, and other defects that could make them break down. For operation-based services, it can be measured by

$$\rho = 1 - \frac{n}{n_s} \tag{2}$$

where $0 \leq \rho \leq 1$ represents reliability; n and n_s denote the number of failed and total operations that occurred in a time interval, respectively. The closer the value of ρ is to 1, the higher the reliability.

Another two measurements of reliability are the *mean time between failures* (MTBF) and the *mean time to failure* (MTTF) [2]. Both MTBF and MTTF can be used for continuously running services. The difference is that the former assumes that a failed system is immediately repaired, whereas the latter assumes that the system is non-repairable. From a service perspective, a measurement from an end-user's viewpoint like the one specified above is better than those from a system perspective like MTBF and MTTF.

As cloud services are based on hardware and software, where unknown defects could occur, consumers value a highly reliable service. In other words, cloud services, ideally, should be fault-free.

4) Responsiveness: Responsiveness (RESP) is the promptness with which cloud services perform a request during a time interval, which can be measured by

$$\tau = 1 - \frac{f_{i=1}^{n}(t_i)}{t_{\max}} \tag{3}$$

where $0 \leq \tau \leq 1$ represents responsiveness, t_i denotes the time between the submission and the completion of the ith request, $t_{\max}$ is a parameter that specifies the *maximum acceptable time* to complete a request ($t_{\max} \geq t_i$), n is the number of requests issued in an operational period, and f is a function that measures the central tendency of a set of data, such as the mean and the median. The closer the value of τ is to 1, the better the responsiveness.

Instead of a *maximum possible time* from a system perspective, we use the maximum acceptable time from an end-user's viewpoint to formulate responsiveness. For cloud services, a measurement from an end-user's viewpoint is better than the one from a system perspective.

Few cloud services currently mention responsiveness in their SLAs. As cloud services advocate computation, in a general meaning, as a service, where a quick response indicates high performance, consumers value an accurate and prompt service.

5) Security: Security (SECY) is the assurance that cloud services are free from viruses, intrusions, spyware, attacks, and other security vulnerabilities that could put them at risk, which can be measured by

$$\theta = 1 - F_T(t) \tag{4}$$

where θ represents security and $F_T(t)$ denotes a cumulative distribution function of a random variable T indicating the time until the first security breach occurs, measured in unit time.

For simplicity, we assume that the security issues occur at random and uniformly in cloud services throughout a time interval. That is, there is no clustering of security issues. The occurrence of security issues, then, can be modeled as a Poisson process with mean $\lambda > 0$. Let a random variable T denote the time from the start of an operational period until the first security breach occurs. T now follows an *exponential distribution* with parameter λ, whose *cumulative distribution function* is described as [13]

$$F_T(t) = 1 - e^{-\lambda t} \quad t \geq 0. \tag{5}$$

Alternatively, its *probability density function* is determined, by differentiation, as

$$f_T(t) = \lambda e^{-\lambda t} \quad t \geq 0. \tag{6}$$

If security issues follow different patterns, we can model them with regression techniques [13]. It should be cautioned here that we do not intend to detect security issues, nor do we intend to enforce security mechanisms for cloud services, which

are beyond the scope of this paper. Rather, we use history information, which is publicly available or can be obtained from a third party, to approximate their security level from an end-user's viewpoint.

Some cloud providers currently claim that their services are secure, but none cover security in their SLAs. As cloud services may store and process users' sensitive data, where security vulnerabilities could occur, consumers value a highly secure service. In other words, cloud services, ideally, should be risk-free.

6) Elasticity: Elasticity (ELAS) is the ability of cloud services to provide resources, in a general sense, on demand during a time interval, which can be measured by

$$\varepsilon = \frac{\sum_{i_1=1}^{n} r_{i_1}}{\sum_{i_2=1}^{n} r_{i_2}} \tag{7}$$

where $0 \leq \varepsilon \leq 1$ represents elasticity, r_{i_1} and r_{i_2} denote the amount of resources allocated and requested in the ith request, respectively, and n is the number of requests issued in an operational period. The closer the value of ε is to 1, the higher the elasticity.

As cloud services promise resources on demand, whereas the amount of physical resources at a specific time is finite, consumers value an elastic service.

C. Discussion

CLOUDQUAL is a quality model that we propose for cloud services. As to its six quality dimensions, five are objective and negotiable, whereas usability remains subjective and nonnegotiable. While some quality dimensions of CLOUDQUAL like availability and reliability are used in other papers, we take a new perspective on them in this paper. Instead of a system perspective assumed in most papers, we regard them from a service perspective, i.e., an end-user's viewpoint, and redefine accordingly the quality dimensions and metrics. So, even if they are not brand-new, they are from a new angle.

IV. Empirical Case Studies

In this section, we first describe three real-world storage clouds, which serve as our case studies. After that, we demonstrate how to use CLOUDQUAL to evaluate their quality. In Section V, we use the three storage clouds to validate CLOUDQUAL.

A. Case Studies

Three well-known storage clouds, namely Amazon S3, Microsoft Windows Azure Blob Storage (Azure Blob), and Aliyun Open Storage Service (Aliyun OSS) are chosen as the case studies [30]. Amazon S3 is an online storage service launched in March 2006 by Amazon.com, Inc. It is used to store and retrieve "any amount of data, at any time, from anywhere on the Web" [30]. Amazon S3 is intended for content storage and distribution, storage for data analysis, backup, archiving, and disaster recovery. It is built to be "secure, reliable, scalable, fast, inexpensive, and simple."

Azure Blob is another online storage service released in February 2010 by Microsoft Corporation. It is used to store "large amounts of unstructured data that can be accessed from anywhere in the world via HTTP or HTTPS" [30]. Azure Blob is intended for serving images or documents directly to a browser, storing files for distributed access, streaming video and audio, performing secure backup and disaster recovery, and storing data for analysis by an on-premise or Windows Azure-hosted service.

Aliyun OSS is yet another online storage service launched in July 2011 by Alibaba Group, whose flagship company—Alibaba.com—is the world's largest online business-to-business trading platform. It offers "large-scale, secure, low-cost, highly reliable storage services", which can be accessed "at any time, in any place" [30]. Aliyun OSS is intended for storing large text, image, video and audio files, and hosting Web sites. It is built to be "secure, reliable, elastic, cost-efficient, and high-performance."

B. Experimental Setup

All experiments are conducted on a Lenovo ThinkCentre desktop with a 2.80-GHz Intel Pentium Dual-Core CPU and a 2.96-GB RAM, running Microsoft Windows 7 Professional Operating System. The machine is located in Kingston, Ontario, Canada. Both Amazon S3 and Azure Blob under test are in the US East, whereas Aliyun OSS is in eastern China. The Internet connection is 100 Mbps. All operations are carried out by invoking the API methods of the three storage clouds with Java programs implemented under NetBeans IDE 6.9.1 with JDK 6u21.

It should be cautioned here that we do know that network distance can have a negative impact on quality dimensions like responsiveness and elasticity, especially for Aliyun OSS. However, for Amazon S3 and Azure Blob, which are both located in Virginia, we assume the impact is minimal and can be ignored. For two reasons, we keep Aliyun OSS in our case studies. First, we want to collect data to validate CLOUDQUAL. Second, from a service perspective, it is the quality delivered, not network distance that matters to an end user. If network distance or performance does have a big impact, then the provider needs to consider moving the "cloud" toward its customers or improve its network performance.

The experiments were conducted between 10 A.M. and 10 P.M. during October 1, 2012 to October 31, 2012. Four operations (i.e., upload, download, update, and delete) are performed each day on Amazon S3, Azure Blob, and Aliyun OSS. The size of files under operation is 1 MB, 10 MB, 100 MB, 1 GB, or 10 GB. For a file of small size, a short operation can take a few seconds to finish, whereas for a file of large size, a long operation may take more than 1 h to complete. For each storage cloud, the testing time is about 3 h, and the total time 9–10 h/day.

C. QoS Evaluation

We use CLOUDQUAL to evaluate the QoS offered by Amazon S3, Azure Blob, and Aliyun OSS. As CLOUDQUAL targets general cloud services, its quality dimensions need to be refined for storage clouds.

1) USAB Evaluation: USAB assesses the ease of use and efficiency of a storage cloud's interface.

At the time of our experiments, Amazon S3 provided both an API for developers and a WUI for Web users. In fact, the Amazon

TABLE I
USABILITY COMPARISONS OF THREE STORAGE CLOUDS

	API	GUI	WUI
Aliyun OSS	Yes	Yes	No
Amazon S3	Yes	No	Yes
Azure Blob	Yes	No	No

Web Services (AWS) Software Developer's Kit (SDK) for Java provides a Java API for Amazon S3. With it, developers can get started in minutes with a single, downloadable package that includes the AWS Java library, code samples, and documentation. Amazon S3 also offers a WUI for Web users. With it, users can easily perform Create, Read, Update, and Delete (CRUD) operations on Amazon S3.

At the time of our experiments, Azure Blob provided an API for developers, but did not offer a WUI or a GUI. Windows Azure SDK for Java provides a Java API for Azure Blob, which is quite similar to AWS SDK for Java. In order to visualize the files stored in Azure Blob, a third-party GUI called Azure Storage Explorer (ASE) can be installed. ASE allows users to perform CRUD operations with a similar level of ease as Amazon S3.

At the time of our experiments, Aliyun OSS provided both an API for developers and a GUI for end users. In fact, Aliyun SDK for Java provides a Java API for Aliyun OSS, which allows users to perform CRUD operations with the same level of enjoyment that Amazon S3 can offer.

In terms of ease of invocation, Amazon S3, Azure Blob, and Aliyun OSS all do a good job. In terms of user-friendliness, Amazon S3 and Aliyun OSS both do a good job, but Aliyun OSS can be improved if it offers a WUI, so that users do not have to install a client GUI on their machine; however, Azure Blob needs to do more work, as it lacks both WUI and GUI. Table I compares usability of Aliyun OSS, Amazon S3, and Azure Blob.

2) AVAL Evaluation: AVAL represents a storage cloud's uptime percentage during a time interval.

Amazon S3, at the point of writing, promises in its SLA "a monthly uptime percentage of at least 99.9% during any monthly billing cycle" [30]. In the event that Amazon S3 does not meet its service commitment, users are eligible to receive a service credit, which "is calculated as a percentage of the total charges paid by users for Amazon S3 for the billing cycle in which the error occurred."

In the experimental period, Amazon S3's uptime and downtime are 31 days and 0 days, respectively. Using (1), its end-to-end availability is calculated as

$$\alpha = \frac{t}{t_s} = \frac{31}{31} = 100.0\%.$$

It should be mentioned that an availability of 100% here does indicate that Amazon S3 fulfills its service commitment, which is 30.97 days available in 31 days. But it does not mean that Amazon S3 is always available in that period, since we observe it for about 3 h/day, which is still a short-time window. However, it is sufficient for the purpose of illustration.

Azure Blob, at the point of writing, does not have an SLA available, and thus we have no knowledge of its promised availability. In the experimental period, Azure Blob's uptime and downtime are 31 days and 0 days, respectively. So, its end-to-end availability is also 100.0%.

Aliyun OSS has an SLA available now, but it does not mention its availability there. As such, we have no information about it. In the experimental period, Aliyun OSS's uptime and downtime are 31 days and 0 days, respectively. So, its end-to-end availability is 100.0% too.

3) REL Evaluation: REL means the assurance that a storage cloud is free from hardware failures, software faults, and network outages.

Errors could occur when we perform CRUD operations on a storage cloud. In fact, we encounter a socket write error, when we upload the same zipped file of 1 GB to Amazon S3 on October 2, 2012. The returned error message says, "The difference between the request time and the current time is too large." Unfortunately, Amazon S3 may not be responsible for such an error, as the error type is labeled as "Client", neither "InternalError" nor "Service-Unavailable" that is stipulated in its SLA. However, such an error could cause users to experience frustration and dissatisfaction.

To stress-test its reliability, in the experimental period, we perform four operations (i.e., upload, download, update, and delete) on Amazon S3 with four zipped files (i.e., 1 MB, 10 MB, 100 MB, and 1 GB files). To our surprise, we encounter at least one failure on 26 of 31 days, when we perform the four operations on the 1 GB file. No failures occur on the other three files. Using (2), the end-to-end reliability of Amazon S3 is determined as

$$\rho = 1 - \frac{n}{n_s} = 1 - \frac{77}{4 \times 4 \times 31} = 84.5\%$$

on the four files, where $n = 77$ and $n_s = 4 \times 4 \times 31$, and 100% on the first three files. It should be noted that a reliability of 84.5% here does not mean that Amazon S3 is responsible for all the errors that occur, since some errors are labeled as the "Client" side.

In our experiments, Azure Blob seems more reliable than Amazon S3. However, we still encounter a failure, when we update a zipped file of 10 GB on Azure Blob on October 3, 2012. The returned message says that an exception occurs without more details about it. As Azure Blob does not have an SLA to stipulate the exception, we are not sure who should be responsible for the failure.

In the experimental period, we perform the four operations on Azure Blob with one additional file (i.e., a 10 GB file). Still, we encounter at least one failure on 14 of 31 days, on the 10 GB file. No failures occur on the other four files. Using (2), the end-to-end reliability of Azure Blob is determined as 94.0% on the five files, where $n = 37$ and $n_s = 4 \times 5 \times 31$, and 100% on the first three and four files.

In our experiments, Aliyun OSS seems less reliable than Amazon S3. In fact, we encounter a failure, when we update a zipped file of 100 MB, on Aliyun OSS on October 1, 2012. The returned message says that a socket write error occurs. As Aliyun OSS does not mention the error in its SLA, we again are not sure who should be responsible for the failure.

In the experimental period, we perform the four operations on Aliyun OSS with one less file (i.e., the 1 GB file). Still, we

encounter at least one failure on 27 of 31 days, on the 100 MB file. No failures occur on the other two files. Using (2), the end-to-end reliability of Aliyun OSS is determined as 76.9% on the three files, where $n = 86$ and $n_s = 4 \times 3 \times 31$.

It should be cautioned here that the failures that occur in the three storage clouds are not from network outages, nor are they from hardware failures. It is the way in which a storage cloud handles a long-running operation that causes the exceptions. So, the failures are from the storage clouds, even if their storage systems do not fail in that period. Fig. 1 compares end-to-end reliability for Aliyun OSS, Amazon S3, and Azure Blob.

4) RESP Evaluation: RESP indicates the promptness with which a storage cloud's user completes a CRUD operation.

Upload, download, and delete operations are examined below to illustrate responsiveness. Assume here that the maximum acceptable time—a predefined parameter—to complete each operation is 500 s on a 10 MB file—a reasonable number we observe in our experiments. First, we perform an upload operation on Amazon S3. In the experimental period, we transfer a zipped file of 10 MB from our desktop to Amazon S3. It takes 38.940 s on average with a standard deviation of 3.565 to complete the operation. Using (3), Amazon S3's end-to-end responsiveness is calculated as

$$\tau = 1 - \frac{t}{t_{\max}} = 1 - \frac{38.940}{500} = 0.922.$$

It should be noted that a responsiveness of 0.922 here may not meet the requirements of some impatient users who cannot tolerate delays of more than 5 s. If that is the case, a faster file uploading method is required.

In our experiments, Azure Blob takes less time to complete the upload operation than Amazon S3. It takes 3.569 s on average with a standard deviation of 0.264 to complete the operation, which is an order of magnitude faster than that of Amazon S3. Using (3), Azure Blob's end-to-end responsiveness is calculated as 0.993. It should be noted that Azure Blob has adopted a new file uploading method—a Silverlight with *TaskParallel Library* (TPL) solution [30]. It can make full use of available network bandwidth, and so is faster and more reliable.

In our experiments, Aliyun OSS spends more time to complete the upload operation than Amazon S3. It takes 318.862 s on average with a standard deviation of 39.371 to complete the operation, which is an order of magnitude slower than that of Amazon S3. Using (3), Aliyun OSS's end-to-end responsiveness is calculated as 0.362.

We also perform a download operation and a delete operation on the 10 MB file on Amazon S3, Azure Blob, and Aliyun OSS. We summarize the results in Table II. Refer to [30] for the details. Fig. 2 shows the end-to-end responsiveness reached in completing the upload, download, and delete operations for the three storage clouds.

In order to provide a whole picture of responsiveness, we vary the size of files to be 1 and 100 MB, and perform the upload, download, and delete operations on Amazon S3, Azure Blob, and Aliyun OSS too. When the size of files increases by an order of magnitude, the time that the three storage clouds spend increases by the same order for the upload operation and the download operation, and the time varies for the delete operation, whereas their end-to-end responsiveness remains almost constant. Refer to [30] for the details.

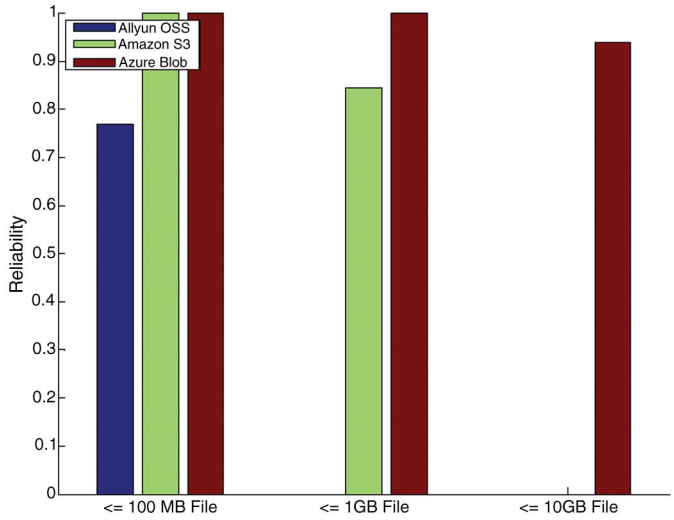

Fig. 1. End-to-end reliability for three storage clouds.

TABLE II
TIME SPENT AND RESPONSIVENESS REACHED OF THREE STORAGE CLOUDS (10 MB)

	Upload[a]		Download		Delete	
	Time	RESP	Time	RESP	Time	RESP
Aliyun OSS	318.862	0.362	33.208	0.934	1.272	0.997
Amazon S3	38.940	0.922	7.254	0.985	0.078	1.000
Azure Blob	3.569	0.993	2.586	0.995	0.078	1.000

[a]For all operations, the file size is 10 MB.

5) SECY Evaluation: SECY denotes the assurance that users' data stored in a storage cloud is under protection and free from data leaks.

In the experimental period, we do not encounter a security breach in Amazon S3, which may not occur or be detected in a short time period. We now illustrate how to model Amazon S3's security. Assume that the time, measured in days, until the first security breach occurs in Amazon S3 can be approximated by a cumulative distribution function

$$F_T(t) = \begin{cases} 0 & t < 0 \\ 1 - e^{-0.0001t} & t \geq 0 \end{cases}. \tag{8}$$

Using (8), the probability that the first security breach occurs in Amazon S3 within 31 days is

$$F_T(31) = P(T \leq 31) = 1 - e^{-0.0031} = 0.003.$$

So, the probability that Amazon S3 is secure is

$$\theta = 1 - F_T(31) = 1 - 0.003 = 0.997.$$

In the experimental period, we do not encounter a security breach in Azure Blob and Aliyun OSS either, and their security could be counted similarly as 99.7% too.

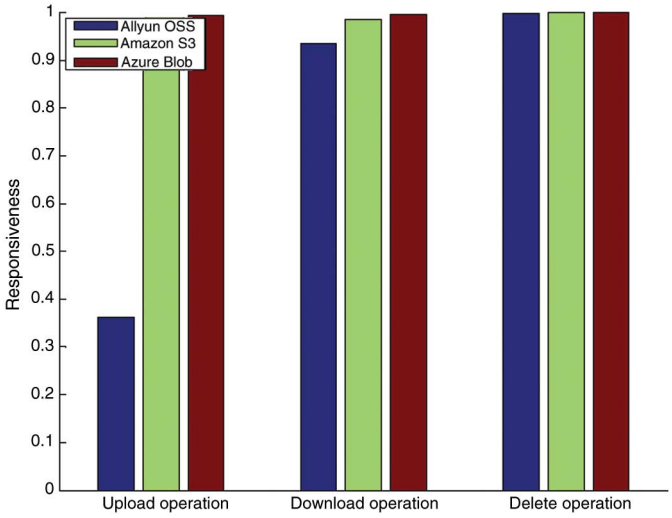

Fig. 2. End-to-end responsiveness comparisons for CRUD operations.

TABLE III
STORAGE REQUESTED AND REACHED IN AMAZON S3

Operation	Storage requested (MB)	Storage reached (MB)	Time to complete (s)
1	1	1	3.976
2	10	10	38.264
3	100	100	362.256
4	1 000	1 000	3 968.706
5	10 000	1 000[a]	>> 4 000[b]

[a,b]Due to time limitations, the two numbers are approximated.

6) ELAS Evaluation: ELAS refers to the ability that a storage cloud can offer storage space on demand in a time interval.

We stress-test Aliyun OSS, Amazon S3, and Azure Blob on October 1, 2012 to roughly determine their *actually reachable elasticity*, i.e., an elasticity that can be reached under a given network bandwidth in a certain time interval. For this purpose, we transfer three to five zipped files of different size from our desktop to each storage cloud in 4000 s, i.e., around 1 h. Due to time limitations and other technical difficulties, we use data obtained in 1 day to give an approximation of elasticity. Table III shows the amount of storage space requested and reached for Amazon S3, and the actual/estimated time to complete the five operations. Using (7), its actual elasticity is determined as

$$\begin{aligned}\varepsilon &= \frac{\sum_{i_1=1}^{n} r_{i_1}}{\sum_{i_2=1}^{n} r_{i_2}} \\ &= \frac{1+10+100+1000+1000}{1+10+100+1000+10\,000} \\ &= 0.190\end{aligned}$$

where $n = 5$. It should be noted that an elasticity of 19.0% here only means Amazon S3's actually reachable elasticity, not what Amazon S3 can reach theoretically. In fact, Amazon S3 claims that it can "store an infinite amount of data in a bucket," which is able to hold as many objects as a user likes, and each object can contain up to 5 TB of data [30].

TABLE IV
STORAGE REQUESTED AND REACHED IN AZURE BLOB

Operation	Storage requested (MB)	Storage reached (MB)	Time to complete (s)
1	1	1	0.547
2	10	10	3.829
3	100	100	30.256
4	1 000	1 000	295.531
5	10 000	4 000[a]	10 613.497

[a]Due to time limitations, the number is approximated.

TABLE V
STORAGE REQUESTED AND REACHED IN ALIYUN OSS

Operation	Storage requested (MB)	Storage reached (MB)	Time to complete (s)
1	1	1	35.236
2	10	10	352.819
3	100	100	3 232.036
4	1 000	100[a]	>> 4 000[b]
5	10 000	100[c]	>> 4 000[d]

[a–d]Due to time limitations, the four numbers are approximated.

In our experiments, Azure Blob can reach a higher elasticity than Amazon S3. Table IV shows the amount of storage space requested and reached for Azure Blob, and the actual/estimated time to complete the five operations. Using (7), its actual elasticity is determined as 46.0%. However, Azure Blob states that a storage account "can contain an unlimited number of containers, as long as their total size is under 100 TB" [30].

In our experiments, Aliyun OSS can only achieve a lower elasticity than Amazon S3. Table V shows the amount of storage space requested and reached for Aliyun OSS, and the actual/estimated time to complete the five operations. Using (7), its actual elasticity is 2.8%. However, Aliyun OSS claims that it can "offer an infinite amount of storage space" [30].

V. METRICS VALIDATION

In this section, we discuss the validation of CLOUDQUAL based on the empirical data presented in Section IV. We specifically want to determine whether our quality model can differentiate service quality, in terms of the quality dimensions and metrics specified in CLOUDQUAL. The technical standard that we adopt to validate CLOUDQUAL is IEEE Std 1061-1998 (R2009) (i.e., *IEEE Standard for a Software Quality Metrics Methodology*). The standard provides "a methodology for establishing quality requirements and identifying, implementing, analyzing, and validating process and product software quality metrics" [9]. It "applies to all software at all phases of any software life cycle," and aims for "measuring or assessing the quality of software." In particular, a metric should have a high degree of association with the quality dimension that it represents in order to be considered valid. As a result, we are concerned with

three validity criteria specified in the standard, i.e., *correlation*, *consistency*, and *discriminative power*. Below, we first detail the three criteria. After that, we illustrate how to validate CLOUDQUAL with them.

A. Validity Criteria

1) Correlation: Let R be the *linear correlation coefficient* between a quality dimension and a metric. The correlation criterion requires that the variation in the quality dimension values explained by the variation in the metric values, i.e., R^2, should exceed a predefined threshold V. In fact, this criterion assesses whether a sufficiently strong linear association exists between a quality dimension and a metric, such that the metric can be used as an indicator for the quality dimension.

2) Consistency: Let r be the *rank correlation coefficient* of paired values of a quality dimension and a metric, where a quality dimension and a metric pair is measured for the same service. The consistency criterion requires that the absolute value of r should exceed a predefined threshold B. In fact, this criterion assesses whether a metric can accurately rank a set of services by a quality dimension.

More specifically, if quality dimension values $D_1, D_2, \ldots, D_n$ of services $1, 2, \ldots, n$, respectively, have the relationship $D_1 > D_2 > \cdots > D_n$, then the relationship $M_1 > M_2 > \cdots > M_n$ should hold for the metric values. In other words, the rank of the metric values should be consistent with that of the quality dimension values.

3) Discriminative Power: Let α be a confidence level. Use a contingency table to hold quality dimension and metric values and compute the chi-square (χ^2) statistic. The discriminative power criterion requires that this value should exceed the chi-square statistic corresponding to α. In fact, this criterion assesses whether a metric can separate a set of high-quality services from a set of low-quality ones for a quality dimension. In other words, the set of metric values of the former should be significantly different from those of the latter.

B. Metrics Validation

The empirical data, which we present in Section IV, is used to validate quality metrics specified in CLOUDQUAL, with the three criteria described above. Here, we show the validation for upload responsiveness, and the remaining operations and quality dimensions are validated in a similar manner [30].

We first validate RESP with respect to the correlation criterion, where V is a threshold of square of the linear correlation coefficient ($0 \leq V \leq 1$). For ease of reference, we show the time that Aliyun OSS, Amazon S3, and Azure Blob spent and the responsiveness reached in completing the upload operation in Table VI, where the file size is 10 MB. From (3), we find that the time metric calculated by function f and the responsiveness are negatively linearly related, and thus their correlation coefficient $R = -1$. It follows that $R^2 = 1$. As $0 \leq V \leq 1$, we always have $R^2 \geq V$, no matter what value V is. So, correlation between the time metric and the responsiveness is confirmed. □

Second, we validate RESP with respect to the consistency criterion, where B is a threshold of rank correlation coefficient ($0 \leq B \leq 1$). *Spearman's rank correlation coefficient*, which assesses how well the relationship between two variables can be described using a monotonic function, is used to determine the rank correlation coefficient between the time metric and the responsiveness. For ease of reference, we show the time that Aliyun OSS, Amazon S3, and Azure Blob spent and the responsiveness reached in completing the upload, download, and delete operations in Table VII, where the file size is 10 MB. For the upload operation, the whole process works as follows.

TABLE VI
TIME SPENT AND RESPONSIVENESS REACHED FOR THREE STORAGE CLOUDS (10 MB)

	Upload[a]	
	Time	RESP
Aliyun OSS	318.862	0.362
Amazon S3	38.940	0.922
Azure Blob	3.569	0.993

[a]For all operations, the file size is 10 MB.

TABLE VII
THE WORKING TABLE TO DETERMINE SPEARMAN'S RANK CORRELATION COEFFICIENT

	Time	RESP	Rank 1	Rank 2	d	d^2
Upload	318.862	0.362	3	3	0	0
	38.940	0.922	2	2	0	0
	3.569	0.993	1	1	0	0

Step 1. Draw a working table, say, Table VII, and fill in the first two columns with pairs of time and responsiveness.

Step 2. In the third and the fourth columns, rank the data in the first and the second columns, respectively (1 for the most valued number, 2 for the second most valued number, etc.). If two or more pieces of data in one column are the same, rank them with the mean of the otherwise ranks.

Step 3. In the fifth column, calculate the difference between the two numbers in each pair of ranks, and then square it in the sixth column.

Step 4. Use (9) to determine Spearman's rank correlation coefficient for the upload operation, where n is the number of pairs of data

$$r = 1 - \frac{6\sum d^2}{n(n^2-1)} = 1 - \frac{6 \times 0}{3 \times (3^2-1)} = 1. \quad (9)$$

As $r = 1$ and $0 \leq B \leq 1$, we always have $r \geq B$, no matter what value B is. So, the time metric can accurately rank a set of services by upload responsiveness. □

Third, we validate RESP with respect to the discriminative power criterion, where α is a threshold of confidence level ($0 \leq a \leq 1$). Let α be 0.10 here—an upper 10-percent point. We hypothesize that a service with responsiveness greater than or equal to 0.950 has a response time less than or equal to 25 s (high responsiveness), and that a service with responsiveness less than 0.950 has a response time greater than 25 s (low responsiveness).

A chi-square (χ^2) statistic, which assesses whether distributions of random variables differ from one another, is used to determine whether the time metric can discriminate a set of services with high-responsiveness from a set of ones with

TABLE VIII
THE CONTINGENCY TABLE TO DETERMINE THE CHI-SQUARE STATISTIC

	Time	High RESP (≥ 0.950)	Low RESP (< 0.950)	Total
Upload	≤ 25	a (1)	b (0)	$a + b$ (1)
	>25	c (0)	d (2)	$c + d$ (2)
	Total	$a + c$ (1)	$b + d$ (2)	$a + b + c + d$ (3)

low-responsiveness. Data from Table VII are used to calculate the chi-square statistic. For the upload operation, the whole process works as follows.

Step 1. Draw a $m \times n$ contingency table, say, Table VIII, where $m = 2$ and $n = 2$ are the number of columns and rows (not including the last ones), respectively.

Step 2. Assign a, b, c, and d with corresponding numbers in Table VIII, i.e., $a = 1$, $b = 0$, $c = 0$, and $d = 2$.

Step 3. Use (10) to determine the chi-square statistic

$$\begin{aligned}\chi^2 &= \frac{(ad - bc)^2(a + b + c + d)}{(a + b)(c + d)(b + d)(a + c)} \\ &= \frac{(2 - 0)^2 \times (1 + 0 + 0 + 2)}{(1 + 0) \times (0 + 2) \times (0 + 2) \times (1 + 0)} \\ &= 3. \end{aligned} \tag{10}$$

The degree of freedom v is determined here as $(m - 1) \times (n - 1) = (2 - 1) \times (2 - 1) = 1$. As $\chi^2_{\alpha,v} = \chi^2_{0.10,1} = 2.71$, we have $\chi^2 > \chi^2_{0.10,1}$. So, the aforementioned hypothesis is confirmed. In terms of the upload operation, the time metric can discriminate a set of services with high-responsiveness from those with low-responsiveness. □

Download and delete responsiveness can be validated in a similar manner. For the reason of conciseness, we omit those parts. Refer to [30] for the validation of AVAL, REL, SECY, and ELAS.

VI. CONCLUSION

Cloud computing is an important component of the backbone of the IoT. Clouds will be required to support large numbers of interactions with varying quality requirements. Service quality will therefore be an important differentiator among cloud providers.

As the spectrum of cloud services expands, how to define and measure their quality becomes an important problem. In this paper, inspired from SERVQUAL and the e-service quality model, we take a service perspective, and initiate a quality model named CLOUDQUAL for cloud services. It is a model that targets general cloud services. CLOUDQUAL comprises six quality dimensions, i.e., usability, availability, reliability, responsiveness, security, and elasticity. A formal specification is given for each quality dimension, and a quality metric is defined for each objective one.

To demonstrate the effectiveness of CLOUDQUAL, we conduct empirical case studies on three storage clouds, i.e., Aliyun OSS, Amazon S3, and Azure Blob. Results show that CLOUDQUAL can evaluate their quality. To demonstrate the soundness of CLOUDQUAL, we validate it with three criteria, i.e., correlation, consistency, and discriminative power. It shows that CLOUDQUAL can differentiate service quality.

REFERENCES

[1] M. Armbrust *et al.*, "A view of cloud computing," *Commun. ACM*, vol. 40, no. 4, pp. 50–58, 2010.

[2] S. D. Conte, H. E. Dunsmore, and V. Y. Shen, *Software Engineering Metrics and Models.* Redwood City, CA, USA: Benjamin Cummings, 1986, pp. 102–105.

[3] D. Durkee, "Why cloud computing will never be free," *Commun. ACM*, vol. 53, no. 5, pp. 62–69, 2010.

[4] S. Ferretti *et al.*, "QoS-aware clouds," in *Proc. 3rd Int. Conf. Cloud Comput.* (CLOUD), Miami, FL, USA, 2010, pp. 321–328.

[5] S. K. Garg, S. Versteeg, and R. Buyya, "A framework for ranking of cloud computing services," *Future Gener. Comput. Syst.*, vol. 29, no. 4, pp. 1012–1023, 2013.

[6] J. Gubbi *et al.*, "Internet of things (IoT): A vision, architectural elements, and future directions," *Future Gener. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, 2013.

[7] W. He and L. Xu, "Integration of distributed enterprise applications: A survey," *IEEE Trans. Ind. Informat.*, vol. 10, no. 1, pp. 35–42, Feb. 2014.

[8] B. Huang, C. Li, and F. Tao, "A chaos control optimal algorithm for QoS-based service composition selection in cloud manufacturing system," *Enterp. Inf. Syst.*, to be published.

[9] *IEEE Standard for a Software Quality Metrics Methodology*, IEEE Std 1061 TM-1998 (R2009), 2009.

[10] Q. Li *et al.*, "Applications integration in a hybrid cloud computing environment: Modelling and platform," *Enterp. Inf. Syst.*, vol. 7, no. 3, pp. 237–271, 2013.

[11] S. Li *et al.*, "Integration of hybrid wireless networks in cloud services oriented enterprise information systems," *Enterp. Inf. Syst.*, vol. 6, no. 2, pp. 165–187, 2012.

[12] S. Li, L. Xu, and X. Wang, "Compressed sensing signal and data acquisition in wireless sensor networks and Internet of things," *IEEE Trans. Ind. Informat.*, vol. 9, no. 4, pp. 2177–2186, Nov. 2013.

[13] D. C. Montgomery and G. C. Runger, *Applied Statistics and Probability for Engineers.* Hoboken, NJ, USA: Wiley, 2011, pp. 132–133.

[14] A. Parasuraman, V. A. Zeithaml, and L. L. Berry, "SERVQUAL: A multiple-item scale for measuring consumer perceptions of service quality," *J. Retailing*, vol. 64, no. 1, pp. 12–40, 1988.

[15] G. Parry, L. Newnes, and X. Huang, "Goods, products and services," in *Service Design and Delivery, Service Science: Research and Innovations in the Service Economy*, M. Macintyre *et al.*, Eds. New York, NY, USA: Springer, 2011, pp. 19–29.

[16] S. I. Swaid and R. T. Wigand, "The customer perspective of E-Service quality: An empirical study," in *Electronic Markets: Benefits, Costs and Risks*, C. Standing, Ed. New York, NY, USA: Palgrave Macmillan, 2009, pp. 36–61.

[17] F. Tao *et al.*, "Research on manufacturing grid resource service optimal-selection and composition framework," *Enterp. Inf. Syst.*, vol. 6, no. 2, pp. 237–264, 2012.

[18] F. Tao *et al.*, "Modelling of combinable relationship-based composition service network and the theoretical proof of its scale-free characteristics," *Enterp. Inf. Syst.*, vol. 6, no. 4, pp. 373–404, 2012.

[19] F. Tao *et al.*, "FC-PACO-RM: A parallel method for service composition optimal-selection in cloud manufacturing system," *IEEE Trans. Ind. Informat.*, vol. 9, no. 4, pp. 2023–2033, Nov. 2013.

[20] J. M. Tien, "Services: A system's perspective," *IEEE Syst. J.*, vol. 2, no. 1, pp. 146–157, Mar. 2008.

[21] L. M. Vaquero *et al.*, "A break in the clouds: Towards a cloud definition," *Comput. Commun. Rev.*, vol. 39, no. 1, pp. 50–55, 2009.

[22] J. W. Wang *et al.*, "On a unified definition of the service systems: What is its identity," *IEEE Syst. J.*, to be published.

[23] J. W. Wang *et al.*, "On domain modelling of the service system with its application to enterprise information systems," *Enterpr. Inf. Syst.*, to be published.

[24] H. Wang, W. He, and F. K. Wang, "Enterprise cloud service architectures," *Inf. Technol. Manag.*, vol. 13, no. 4, pp. 445–454, 2012.

[25] B. M. Wilamowski, R. C. Jaeger, and M. O. Kaynak, "Neuro-fuzzy architecture for CMOS implementation," *IEEE Trans. Ind. Electron.*, vol. 46, no. 6, pp. 1132–1136, Dec. 1999.

[26] B. M. Wilamowski and O. Kaynak, "Oil well diagnosis by sensing terminal characteristics of the induction motor," *IEEE Trans. Ind. Electron.*, vol. 47, no. 5, pp. 1100–1107, Oct. 2000.
[27] L. Xu, "Enterprise systems: State-of-the-art and future trends," *IEEE Trans. Ind. Informat.*, vol. 7, no. 4, pp. 630–640, Nov. 2011.
[28] L. Xu, "Guest editorial: Advances of systems research in service industry," *IEEE Syst. J.*, to be published.
[29] L. Xu, "Introduction: Systems science in industrial sectors," *Syst. Res. Behav. Sci.*, vol. 30, no. 3, pp. 211–213, 2013.
[30] X. Zheng, "Qos representation, negotiation and assurance in cloud services," Ph.D dissertation, School of Computing, Queen's University, Kingston, Ontario, Canada, 2014.

Xianrong Zheng (S'13), photograph and biography not available at the time of publication.

Patrick Martin, photograph and biography not available at the time of publication.

Kathryn Brohman, photograph and biography not available at the time of publication.

Li Da Xu (M'86–SM'11), photograph and biography not available at the time of publication.