1. INTRODUCTION

As a convenient and on-demand IT service delivery paradigm, cloud computing has dramatically changed how business applications are built and run. For service users, it eliminates the costs and complexity of buying, configuring, and managing the hardware and software needed for applications (Armbrust et al., 2009).

Instead, these computing resources and management capabilities are delivered as a service over the Internet. According to the type of service provided, cloud computing can be divided into the following three categories: Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) (Mell & Grance, 2009).

Among these three categories, PaaS is becoming a popular one in the current cloud computing practice (Lawton, 2008). It allows
PaaS consumers (usually web application developers) to build, deploy, and manage custom applications on PaaS. PaaS consumers do not need to control the underlying cloud infrastructure including network, servers, operating systems, or storage, but can monitor and control the deployed applications (Mell & Grance, 2009). Application developers thus can focus on building new applications, adding new functionality to existing applications, and fixing bugs in applications, without worrying about how to configure or manage the runtime environment of applications. There is no doubt that PaaS can help application developers to increase their productivity of development related activities by letting them focus on jobs they are proficient at. Additionally, the dynamic scaling ability provided by PaaS can help PaaS consumers reduce their cost and improve the Quality of Service (QoS) of applications hosted on PaaS, which is another reason for application developers to choose PaaS.

Of course, the burden of configuring and managing the computing resources has not disappeared, and most of it is moved onto the shoulders of PaaS providers. They are expected to serve each of their consumers with on-demand service. When the number of hosted applications and the complexity of the whole system grow, it is a serious challenge for PaaS providers to deliver satisfied services as they promised. How to manage the computing resources to ensure the quality of service provided by PaaS becomes an important factor in the success of a PaaS system.

Monitoring is an essential part of PaaS management. Basically, PaaS providers need to monitor whether applications running on PaaS are in good quality. The performance of each application is evaluated against QoS attributes defined in Service Level Agreement (SLA) (Patel, Ranabahu, & Sheth, 2009), and violation of SLA may bring penalty to PaaS providers. Meanwhile, PaaS need to monitor different kinds of resource, so as to make adjustment actions automatically or manually. One characteristic of cloud computing is that computing resources are assigned dynamically according to each application’s demand. This may give consumers the illusion that they can get infinite resources. Actually, the overall computing resource of a PaaS is always limited, especially from the perspective of PaaS providers. Consequently, PaaS providers need to adjust their resource allocation based on current monitored resource utilization and application performance. In some PaaS like Google App Engine (Google, 2011), each consumer is constrained with some quota of computing resource usage. In such case, PaaS providers also need to monitor the aggregate resource consumption of each application over a given period to see if the resource consumption is within its quota. Additionally, monitoring is also an essential service needed by PaaS consumers. Although PaaS consumers do not need to worry about how to operate and manage the execution environment of their applications, they still need to get some runtime statistics of their applications, which can be leveraged to help them upgrade their applications or to help them choose an appropriate quota.

Monitoring in PaaS is a more challenging issue than the one in a single application. The diversity and heterogeneity of software components in PaaS makes it difficult to implement monitoring. For example, in order to serve web applications of various types, application servers of different versions from different vendors should be provided to maintain compatibility. Each of them may have their own monitoring and controlling API, and these APIs may provide runtime data in different format, which may be inconsistent with the performance metric and resource consumption metric that PaaS providers need. Additionally, PaaS consumers may also ask for their own monitoring view which includes custom monitoring metrics. Consequently, it is a time consuming task to implement monitoring for each target and for each view, and it is hard to make adjustment manually or automatically based on the raw runtime data.

In this paper, we propose a model-based monitoring and controlling approach to support PaaS management. A high level monitoring and
Tracing the Implementation of Non-Functional Requirements
www.igi-global.com/chapter/tracing-implementation-non-functional-requirements/52227?camid=4v1a

Service Quality Evaluation Method for Community-Based Software Outsourcing Process
www.igi-global.com/article/service-quality-evaluation-method-community/53131?camid=4v1a