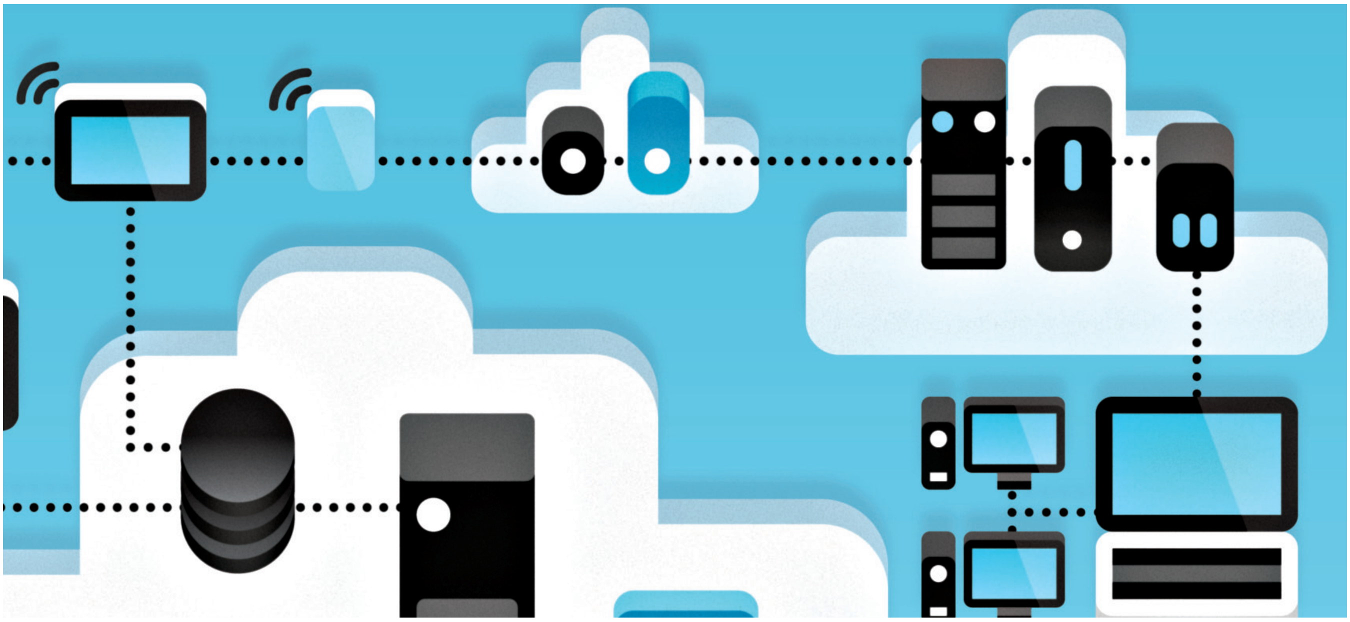


Software Engineering for the Cloud

John Grundy, Swinburne University of Technology

Gerald Kaefer, Siemens AG

Jacky Keong, Hong Kong Polytechnic University

Anna Liu, National ICT Australia

FOR DECADES, WHEN organizations needed to increase their computer systems' data and computation capacity, they had two options: buy more hardware or make the IT operation more efficient. Either choice also implied reengineering the related software—often substantially, in most cases—to replicate databases, scale processing capability, and support a greater number of concurrent processes and users. Along with these hardware and software costs came increased deployment and maintenance costs, both for the computer software and its dedicated hardware and network infrastructure. These "oversized" systems often featured redundant capacity to ensure that they could handle peak demands, but they went underutilized at other times. From a consumer's perspective, IT as a "utility"—you buy more or less of it as you need it, just like other utilities—is the best option.

Cloud computing, as defined by the US National Institute of Standards and Technology, offers a drastically different approach to IT resource delivery, allowing users to lease data and processing capacity from a "cloud"

0740-7459/12/$31.00 © 2012 IEEE

(pool) of interconnected computing systems, maintained by someone else and shared by others, to the level required (http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf). This model has rapidly become a computing paradigm of great interest to the software community (www.gartner.com/DisplayDocument?doc_cd=166525&ref=g_SiteLink). Several cloud computing providers now exist, offering a complex "compute, data storage, and communication for hire" model. The potential benefits of this approach to service delivery include a focus on core business (software service delivery versus platforms), a rental model (versus purchase and maintain), and platform provider-maintained capabilities that might otherwise be challenging (for example, pay-per-use of service, service integration, security systems, and hardware and OS maintenance).

Given the current interest and demand in trying to use the cloud computing model in the software practitioner community, we strongly felt it was time for an *IEEE Software* special issue focusing on software engineering for cloud computing.

Many Benefits, Many Issues

Cloud computing benefits such as agility, elasticity, availability, and cost efficiency require software engineered for cloud platforms. However, due to the concept's recent development, many open issues surround the use of software services from the cloud:

- software requirements engineering for cloud services;
- software architecting and design for cloud services;
- testing cloud services;
- ensuring quality attributes are met for cloud services, such as performance, scalability, security, and availability;
- development methods and tools for engineering cloud services;
- service platforms and composition approaches;
- project management approaches for cloud services and applications; and
- cloud deployment options and justifications.

Independently, each of these aspects must be addressed when engineering practical cloud-based services and solutions today, especially if services are intended for use in an industrial environment.

Requirements

Gathering requirements for cloud services is a rich research area that hasn't yet received much attention. Besides the traditional requirements engineering challenges, this task has unique challenges in cloud projects. For example, in the acquisition phase, organizations must gain a better understanding of their real needs for the cloud—besides top-level business goals such as reducing cost and complexity, are they looking for a platform, infrastructure, or software as a service offering (PaaS, IaaS, and SaaS)? How would they go about determining this? What about determining the organizational and regulatory needs to derive the preferred cloud model (public, private, hybrid, or community cloud)? How can organizations weigh the true opportunities versus risks, and the costs versus the benefits?

Next come the "-ilities," such as scalability, reliability, security, and portability. How does an organization engineer its service requirements in terms of security, privacy, and performance dimensions? What about service-level agreement (SLA) requirements? How does an organization forecast in terms of resource utilization for the future when it finds itself in a much more dynamic environment such as the cloud? How would it determine pricing differently?

Architecture

One of the key values of the cloud infrastructure is its ability to serve many millions of customers around the world at any point in time. To achieve this, multitenant applications are a must; so is the provisioning of robust metering and billing services.

Typical cloud platforms exhibit several unique features, including large-scale replication strategies, highly parallel queries, geographically distributed components, and disaster recovery. Cloud platforms also provide services for mobile device integration and hybrid cloud coupling, all of which call for sophisticated architecture design methods as well as proven design patterns for the cloud computing environment.

> Cloud computing benefits such as agility, elasticity, availability, and cost efficiency require software engineered for cloud platforms.

Testing

Cloud computing platforms are largely opaque entities, meaning that there's little visibility into how runtime components work. Virtualization/virtual machine layers, middleware, multitenancy, and high-availability support further complicate this picture, requiring a heightened need for novel testing and monitoring approaches. The cloud also presents an opportunity to do intensive load testing in a highly parallel manner, which could potentially improve the state of practice in terms of load and regression testing methodologies.

Quality

In using cloud services, consumers cede control—they have less visibility into what happens at runtime and less ability to fix things themselves should anything go wrong.

Performance measurements, security conformance evaluations, and assessment reliability determinations are much more difficult in the cloud, primarily because of its large variation in results over time. The large set of environmental factors, such as network conditions, other users' behavior, and cloud provider configurations, in particular, vary over time.

Development Methods

Development in a cloud environment is a very broad, interdisciplinary challenge—developing cloud computing solutions on the IaaS layer is closely related to IT infrastructure design and integration, whereas developing on the PaaS layer seems like a developer playground. In reality, developers must understand both worlds in order to figure out the best layered design.

Currently, different platforms provide their own development environment and tools for engineering services in their specific cloud. The Google App Engine (http://appengine.google.com), for example, supports various Python- and Java-based tools, whereas Microsoft's Azure platform (www.microsoft.com/windowsazure) supports the .NET framework with Visual Studio.

Model-driven development that specifically focuses on common design/building blocks across cloud platforms and layers holds some promise in improving the current state of development methods and tools; likewise, annotation approaches that give developers greater control over mapping requirements to target cloud platforms could offer beneficial advances in facilitating cross-platform cloud service engineering.

Composition Approaches

The service-oriented architecture (SOA) approach has been on a steadily increasing adoption curve for several years, but not all traditional SOA techniques are applicable to the cloud environment. A major difference is that a SOA design for the cloud generally can't start entirely from scratch. Typically, cloud platforms need to establish ecosystems of collaborating services, some existent and some new—for example, composite search, order, payment, shipping, human resources, and audit. This new cloud environment calls for additional service composition principles. As an example, NOSQL is a popular cloud database model due to its high horizontal scalability feature, but composition across application services and data services (built on NOSQL databases) will require new ways of optimizing for performance as well as data consistency.

Additional questions arise: How can an organization orchestrate business processes across both local services and services in the cloud? Which cloud services will be required? Can traditional workflow or orchestration technologies apply? What about security issues when crossing the boundaries?

Project Management

Enterprise organizations often ask about governance issues in cloud computing: How is SLA managed? How much visibility is there into the running of a business service inside a cloud? How do you know whether an SLA was met—and what happens when it isn't? Little published guidance or experience in cloud vendor management, including exit conditions, currently exists.

The use of cloud computing could have a major impact on an enterprise's business architecture, especially when delivering SaaS; it will also alter the roles and responsibilities of various IT professionals. For example, the cloud can provision developer and test machines much faster via its dynamic resource provisioning capability, so deployment processes might require a new approach.

Cloud Deployment Strategy

Deploying and migrating software applications to the cloud involves many combinations of options that vary widely in their characteristics and performances, from different combinations of CPU, memory, storage, and network options to IT resource management services, to algorithms that can perform dynamic resource scaling.

Developing software systems and choosing the most suitable cloud deployment option using these heterogeneous resources is both nontrivial and difficult to justify for at least two reasons. First, it's often impossible to acquire all the infrastructure resources required to evaluate the performance and cost implications. Even if all the resources are available for such an evaluation, identifying different deployment options and managing them properly for benchmarking experiments is unwieldy and time-consuming. Second, cloud computing applications are specific to certain computing environments and subject to different deployment models, such as IaaS, PaaS, and SaaS. These different models fundamentally change resource utilization strategies on different system levels as well as the cloud system architecture required to deploy the application.

In This Issue

Out of 13 articles originally submitted on this topic, we chose four articles that represent a range of issues as outlined earlier. In particular, these selected articles focus on software processes suitable for cloud computing, cloud architectures, design of cloud-based applications and algorithms, composition of cloud-based services, testing cloud-based applications and services, quality and security aspects of cloud-hosted services, and automated engineering for cloud applications and services.

In "Environment Modeling for Automated Testing of Cloud Applications," Linghao Zhang and colleagues describe a new environment for modeling automated tests targeted specifically at cloud-hosted applications. Their approach allows developers to exhaustively test cloud applications hosted on the Azure platform, and their work advances support for cloud development processes, automated software engineering techniques for the cloud, and cloud application quality of service.

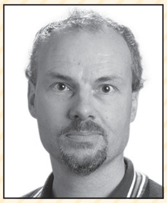
In "A Distributed Access Control Architecture for Cloud Computing," Abdulrahman Almutairi and colleagues describe a new distributed access control architecture for cloud applications. They apply principles from security engineering to multitenant cloud applications to support robust authorization of service access. Their work contributes to cloud-based application architecting, nonfunctional requirements, and security engineering.

In "Testing in the Cloud: Exploring the Practice," Leah Riungu-Kalliosaari and colleagues present the results of a survey of eight organizations using the cloud and their testing approaches and processes. Their focus is on use of cloud testing toolkits and associated processes, and it highlights several current practices and opportunities. Their work also contributes to our understanding of current cloud testing practices, support tools, software processes for cloud applications, and software tools for cloud applications.

Finally, in "Evaluating High-Performance Computing on Google App Engine," Radu Prodan and colleagues present an evaluation of the Google-AppEngine's potential for supporting large-scale parallel computing applications. They run a set of parallel computations on the Google cloud infrastructure and assess various characteristics of this platform for highly scalable computational systems. Their work contributes to cloud-based application requirements, design, technology usage, scaling and performance, and use of empirical experimentation.

We hope that these articles will be of interest and assistance to both software practitioners working on their own cloud-based services and applications but also to those considering a move to the cloud. Cloud computing appears to be a major paradigm shift for 21st century computing, and we expect many more innovations—and challenges—in software engineering for the cloud.

ABOUT THE AUTHORS



JOHN GRUNDY is a professor of software engineering and currently head of computer science and software engineering and director of the Centre for Computing and Engineering Software Systems at the Swinburne University of Technology, Melbourne, Australia. His research interests include software engineering methods and tools, model-driven software engineering, and software architecture. Grundy has a PhD in computer science from the University of Auckland, New Zealand. He is a member of IEEE, IEEE Computer Society, ACM, and a member of the *IEEE Software* Editorial Board. Contact him at jgrundy@swin.edu.au.

GERALD KAEFER is a program manager for cloud computing at Siemens AG. His research interests include the strategy, business, and technology impact of cloud platforms and cloud computing architecture. Kaefer has a PhD in computer engineering from Graz University of Technology. He is a member of IEEE and IEEE Computer Society. Contact him at gerald.kaefer@siemens.com.

JACKY KEUNG is an assistant professor in the Department of Computing at the Hong Kong Polytechnic University. His research interests include cloud computing, empirical software engineering, machine learning, data-intensive analysis, software measurement and its applications to cloud computing, and software cost estimation. Keung has a PhD from the University of New South Wales. He is a member of the Australian Computer Society, ACM, and IEEE Computer Society. Contact him at jacky.keung@comp.polyu.edu.hk.

ANNA LIU is a research group leader for software systems at National ICT Australia. Her research interests are primarily in software engineering and distributed systems, particularly design and development techniques for performance, scale, and elasticity. She serves as a steering committee member for the Australian Software Engineering Conference series. Contact her at anna.liu@nicta.com.au.