Chapter 88
Cloud-Based Testing for Context-Aware Cyber-Physical Systems

Christian Berger
University of Gothenburg, Sweden

ABSTRACT

Cloud-based applications like email services or office suites enable real-time collaboration and traceability for shared data from nearly anywhere by using a modern web-browser. Thus, a significant shift has happened to these common applications to focus only on their usage than on their maintenance. However, today's software development projects spend a noteworthy amount of resources to setup and maintain necessary development tools–over and over again. Thus, a similar shift for these development tools in the future would enable to spend valuable resources more on the actual project's goals than on the tools' maintenance. Especially development projects for cyber-physical systems, which interact with the real life's surroundings by relying on sensors and actuators, have specific needs when using cloud-based solutions. In this contribution, preconditions, design decisions, and limitations of a cloud-based testing approach for CPS are outlined and discussed on the example "Hesperia." "Hesperia" bases on the experiences from the development of "Caroline"–an autonomously driving vehicle for the 2007 DARPA Urban Challenge. "Hesperia" as a cloud-based testing approach was tested 2009 during the development of an autonomously driving vehicle at the University of California, Berkeley.

INTRODUCTION

Nowadays, cloud-based applications are *in vogue* for services, which were operated separately and individually in the past. Cloud computing bases on the fundamental idea that today's servers are sufficiently powerful so that the underlying hardware can be entirely encapsulated through special CPU instructions. Thus, it is possible to run several instances of any major operating system (OS) independently without significant losses in their overall performance. This basic concept is called Infrastructure-as-a-Service (IaaS).

DOI: 10.4018/978-1-4666-4301-7.ch088

The separation of the OSes from the hardware CPU enables the migration from one real computation node to another node at run-time. Thus, the underlying computing power can be extended or reduced to match the required computation load–manually or even automatically as a software layer in terms of Platform-as-a-Service (PaaS). Therefore, the applications, which are running inside this environment, can react in an elastic manner on changing system loads.

Very popular applications are web-based email solutions or entire office suites, which are running on top of the PaaS. These applications realize the Software-as-a-Service (SaaS) principle. Furthermore, cloud-based office solutions often provide collaborative features to track changes or to work simultaneously. Thus, traditional workflows are not only simplified but also accelerated by using SaaS in public or private clouds.

However, the most popular experience for cloud-based solutions is focusing on end-user-related services. But, how can the modern software development benefit from those cloud-based approaches? To answer this question, the processes of today's software engineering must be analyzed to identify potential use cases. Some classical services, which are mandatory for any software development project nowadays, could already be successfully migrated into the cloud. Such services are for example a software artifacts' versioning system to track changes over time; furthermore, "build farms" are also available where computing nodes are compiling a set of source files with different configurations or even on different OSes and platforms to identify interoperability errors. Both services were successfully realized for example at "SourceForge" (Korecki, 2005) or by the "openSUSE Build Service" (Schroeter, 2009).

Nevertheless, the aforementioned services can be regarded as the backbone of a modern software development project for which a fully cloud-based solution is not necessarily required (low/none cloud-based solution). The other extreme is the developer's experience at the front-end site: Mandatory tools for editing source code are directly migrated into the cloud like Cloud9 IDE for JavaScript editing (von Wedel, 2011). These solutions also allow the execution of the created artifacts (full cloud-based solution). Thus, any developer can participate by using a modern web browser only.

The former situation is the minimal tool support for a collaborative software development project and the latter is a cloud-based-only solution for a very specific problem domain. Between both extremes of cloud-based support, the cloud's potential must be evaluated for today's software engineering projects of any industrial sector. While there are projects whose development and especially testing processes can already migrated into the cloud with manageable effort, there are many other development projects whose software-driven functions rely mainly on sensors to perceive information about their surroundings to perform actions. These systems are called cyber-physical systems (CPS) whose required computation time is not an issue of performance but of correctness (Lee & Seshia, 2011). For using the cloud to support testing processes of these CPS for example, special requirements and conditions must be kept in mind.

On the example of car manufacturers, which are increasingly equipping their vehicles with software-intense functions to protect passengers, pedestrians, and bicyclists by using cameras, radars, and the like, their specific testing processes must be revised to be ready for the cloud. However, these processes base mainly on real vehicle test-drives, which are insufficient due to the following circumstances:

- The test processes depend on the system's surroundings, i.e. various, representable, and repeatable context situations are necessarily required.
- The system's overall complexity is still increasing for which an also increasing testing effort is mandatory.