Cloud testing and support to the software lifecycle

Elisabetta Di Nitto
The software lifecycle and the cloud

- PaaS/IaaS approaches seen so far mostly dedicated to offer run-time features
- Google App Engine and MS Azure also offer programming features
  - IDEs
  - Programming models (e.g., web role/worker role in Azure, parallelism managed through task queues in GAE)
- Coverage of other aspects of the software lifecycle
  - Cloud testing and testing Cloud applications
    - CloudBees
CLOUD TESTING
Software testing is the process of assessing, through execution, the functional and non-functional characteristics of a program.

For large systems:
- High number of test cases (thousands)
- Need to execute the same tests several times during the day
- Need to handle testing for multiple configurations (e.g., multiple browsers, various operating systems, various versions/types of databases, …)
## Software testing and the Cloud

<table>
<thead>
<tr>
<th>Cloud characteristics</th>
<th>Testing problems</th>
</tr>
</thead>
<tbody>
<tr>
<td>Possibility to have a large number of parallel executions</td>
<td>➔ Large test suites</td>
</tr>
<tr>
<td>Possibility to exploit resources (and pay for them) only when needed</td>
<td>➔ Execute tests along the day</td>
</tr>
<tr>
<td>Possibility to build virtual machines each one with a different configuration</td>
<td>➔ Handle different configurations</td>
</tr>
</tbody>
</table>
Cloud testing

- Cloud testing means using the Cloud to support the testing phase
- It seems to be the silver bullet, but it requires care!
Risks of Cloud testing

- Testing artifacts need to be migrated to the cloud even though still kept in sync with other development activities.

- Artifacts to be migrated
  - Test cases/test suites
  - Scaffolding
  - Software to be tested
  - Testing tools

- Specific Cloud issues, e.g., security, should be taken into account.

- Test code, libraries, dependencies may not be supported by the selected Cloud.
Characteristics of applications that can be tested in the Cloud

- Test case independence
  - If test cases are temporally dependent from each other, they cannot be run in parallel

- Standard operating environment
  - Otherwise incompatibilities may arise

- Availability of programmatic interfaces
  - GUI-based testing could be less convenient on the Cloud
Types of testing suitable for the Cloud

- Unit testing
  - Easy to automatize
  - A killer application of Cloud testing

- High Volume Automated Test
  - Looks for failures that happen only after a prolonged test (e.g., deadlock, race conditions, …)
  - Resource intensive
  - It can rely on the large number of resources offered by the Cloud

- Performance testing
  - Used to determine application throughput, resolving concurrency issues, tuning server response time, assessing availability
  - In a Cloud environment it is possible to emulate load on demand
What do we install on the Cloud?

- The entire system to be tested, together with all it is needed for performing testing
  - This is what we have assumed so far

- We use the Cloud for hosting/emulating part of the system
Examples of testing tool: D-Cloud

- Support execution of tests with fault injection

---

<table>
<thead>
<tr>
<th>device</th>
<th>contents</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hard disk</td>
<td>Error of specified sector</td>
<td>badblock</td>
</tr>
<tr>
<td></td>
<td>Specified sector is read-only</td>
<td>readonly</td>
</tr>
<tr>
<td></td>
<td>Error detection by ECC</td>
<td>ecc</td>
</tr>
<tr>
<td></td>
<td>Received data contains error</td>
<td>corrupt</td>
</tr>
<tr>
<td></td>
<td>Response of disk becomes slow</td>
<td>slow</td>
</tr>
<tr>
<td>Network</td>
<td>1bit error of packet</td>
<td>1bit</td>
</tr>
<tr>
<td></td>
<td>2bit error of packet</td>
<td>2bit</td>
</tr>
<tr>
<td></td>
<td>Error detection by CRC</td>
<td>crc</td>
</tr>
<tr>
<td></td>
<td>Packet loss</td>
<td>loss</td>
</tr>
<tr>
<td></td>
<td>NIC is not responding</td>
<td>nic</td>
</tr>
<tr>
<td>Memory</td>
<td>Bit error</td>
<td>bit</td>
</tr>
<tr>
<td></td>
<td>Byte at specified address contains error</td>
<td>byte</td>
</tr>
</tbody>
</table>

---
Using the Cloud for emulating part of the system

- Emulation of large traffic in stress and performance testing
- Application-Behavior Virtualization to replace expensive/limited software
SOASTA CloudTest

- Use the Cloud to inject large traffic hitting the application under test
- Testing can happen also in production
Application-Behavior Virtualization

- Focuses on enabling testing when the software under test includes
  - Missing/unstable components
  - Non fully accessible third-party/partner systems and services
  - Systems that are too complex for test labs
- Enables testers to rapidly isolate and virtualize just the behavior of the specific dependent components that they need to exercise in order to complete their end-to-end transactions.
Application-Behavior Virtualization

Steps

- Identify the components to virtualize
- Capture the behavior of the associated transactions, messages, services in virtual assets
- Configure the virtual assets parameterizing conditional behavior, performance criteria, test data
- Use the virtual assets to emulate the actual behavior of components even when they are not available
System Under Test

System behavior and data captured

Dependent Architecture
# Test environments in the Cloud (1)

<table>
<thead>
<tr>
<th>Attributes of Cloud solutions</th>
<th>Characteristics</th>
<th>Benefits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advanced virtualization</td>
<td>Test resources (infra, tools and people) are pooled and virtualized</td>
<td>Providing efficient implementation of independent infrastructure</td>
</tr>
<tr>
<td>Rapid Provisioning</td>
<td>Test resources are provisioned on demand</td>
<td>Reducing test setup and execution time and eliminating errors</td>
</tr>
<tr>
<td>Service Catalog ordering</td>
<td>Test environment are readily available</td>
<td>Enabling visibility, control and automation</td>
</tr>
<tr>
<td>Elastic scaling</td>
<td>Test environment can be scaled up or down by large factor as the need emerges</td>
<td>Optimizes, infra and software license usage</td>
</tr>
<tr>
<td>Flexible pricing</td>
<td>Test resources are priced on supported topology and project phases</td>
<td>Offering pricing options tailored to user resource need</td>
</tr>
<tr>
<td>Metering and billing</td>
<td>Test resources used in reserved are charged back to LOBs</td>
<td>Prioritizing innovative projects</td>
</tr>
</tbody>
</table>
### Attributes of cloud solutions

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Benefits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Platform based configurations (for testing) can be created and managed</td>
<td>Comprehensive assurance on product / service</td>
</tr>
<tr>
<td>Release based configurations (for testing) can be created and managed</td>
<td>Complete assurance on maintenance of product / service</td>
</tr>
<tr>
<td>Emerging alternative models for supporting client testing needs</td>
<td>Collaborative platform enables collective productivity</td>
</tr>
<tr>
<td>In a multi component architecture, availability of a dependant component(s) managed for testing</td>
<td>Effective completion of component level testing, despite the dependencies on Critical components</td>
</tr>
<tr>
<td>In a multi component architecture, availability of a dependant component(s) managed for testing</td>
<td>Effective completion of component level testing, despite the dependencies on Critical components</td>
</tr>
</tbody>
</table>
# Test environments in the Cloud (3)

<table>
<thead>
<tr>
<th>Capability</th>
<th>Past</th>
<th>Present and future</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server / Storage utilization</td>
<td>10-20%</td>
<td>70-90%</td>
</tr>
<tr>
<td>Cost Reduction</td>
<td>Nil</td>
<td>20-30%</td>
</tr>
<tr>
<td>Self Service</td>
<td>None</td>
<td>Unlimited</td>
</tr>
<tr>
<td>Test Provisioning</td>
<td>Weeks</td>
<td>Minutes</td>
</tr>
<tr>
<td>Change Management</td>
<td>Months</td>
<td>Days / Hours</td>
</tr>
<tr>
<td>Release Management</td>
<td>Weeks</td>
<td>Minutes</td>
</tr>
<tr>
<td>Metering / Billing</td>
<td>Fixed cost model</td>
<td>Granular</td>
</tr>
<tr>
<td>Re-platform ability (Compatibility Testing)</td>
<td>Prohibitively expensive</td>
<td>Engineering possibility with affordable cost</td>
</tr>
<tr>
<td>Maintenance of multiple test best beds for multiple release testing</td>
<td>Prohibitively expensive</td>
<td>Engineering possibility with affordable cost</td>
</tr>
<tr>
<td>Multiple Tools testing (Application Security – for false positives – false negatives triangulations)</td>
<td>Prohibitively expensive</td>
<td>Affordable because of utility pricing and improved coverage</td>
</tr>
<tr>
<td>New Alternate for BCP (Business Continuity Plan) and DR (Disaster Recovery)</td>
<td>Not Applicable</td>
<td>A Distinct possibility</td>
</tr>
<tr>
<td>Test factories and TCoE Setup for clients</td>
<td>8-12 months</td>
<td>3-6 months</td>
</tr>
</tbody>
</table>
Testing as a Service (TaaS)

Potential benefits

- Tool license and infrastructure costs on a pay-per-use basis
- Flexibility and wide range
- No setup and procurement time wastage
- No need of hiring expert people
The landscape of current offering [CSC]

Platforms
- Windows (Multilingual, Multi CPUs)
- MS-SQL, Oracle, DB2, Informix, MySQL, Teradata
- Sun Solaris
- Linux (Red hat, Linux, Debian)

Cloud Based Testing Service Offerings under TaaS Ecosystem

- Platforms
  - Windows (Multilingual, Multi CPUs)
  - MS-SQL, Oracle, DB2, Informix, MySQL, Teradata
  - Linux (Red hat, Linux, Debian)
  - Sun Solaris

- Elastic servers
  - (Tomcat, My SQL combination s)

FUTURE OFFERINGS

- Cloud Based Testing Service Offerings under TaaS Ecosystem
- Industrialized Solution – CAT / SAPT
- CSC OPEN SOURCE CLOUD
- CSC TRUSTED CLOUD

Apps & Services
- Client / Server
- Enterprise Applications – SAP, Oracle, Cadence, PTC, Sugar CRM

Tools Sandbox
- Web Application Performance Testing
- Web Applications

Tools
- CloudTest by SOASTA
- CSC Trust Cloud Powered by Skytap

CSC OPEN SOURCE CLOUD

CSC TRUSTED CLOUD

Industrialized Solution – CAT / SAPT

Enterprise Applications – SAP, Oracle, Cadence, PTC, Sugar CRM

Tools
- CloudTest by SOASTA
- CSC Trust Cloud Powered by Skytap

Industrialized Solution – CAT / SAPT

Enterprise Applications – SAP, Oracle, Cadence, PTC, Sugar CRM

Tools
- CloudTest by SOASTA
- CSC Trust Cloud Powered by Skytap

Industrialized Solution – CAT / SAPT

Enterprise Applications – SAP, Oracle, Cadence, PTC, Sugar CRM

Tools
- CloudTest by SOASTA
- CSC Trust Cloud Powered by Skytap
TaaS experience at CSC

Customer requests Testing from CSC
Online portal / Telesales

Provisioning / Scheduling of Tests

CSC conducts Testing

CSC publishes Test Results & Reports

CSC On-Demand Testing in Cloud

Client

Test Requirements & Application Components

Customer Sign-off

Certify Application

Test Summary Reports

Collaboration

Test Execution

Infrastructur Stabilization

CSC

Cloud Orchestration

Test Strategy & Planning

Virtual Test lab using SkyTap

SOASTA Cloud

Automated Software Quality Tools

Application Under Test

Web Server

Application Servers

Database servers

CSC & Client

CSC & Client
TESTING CLOUD-BASED APPLICATIONS
What things are the same about testing cloud solutions?


- Requirements coverage based testing
- Risk-based structured test approach
- Test Plan / Test Phases
- Test Cases / Test Data / Test Automation
- Defect Management / Functional Test
- Use of off-shore resources / Virtualization
- Politics {Quality / Schedule / Resources}
What is different about testing cloud solutions?


- Shared multi-tenant test environments
- Security (Test User IDs / SSO)
- Integration of on/off premise systems
- Performance / volume test
- Defect isolation
- Documentation of “Out Of the Box” reqs
- Release Management
What is harder than expected?

- Isolating the architectural components for a performance test
- Understanding that a cloud solution vendor delivers new releases which you have to take!
- Accepting that you have a “black box” in your environment
- Determining who owns defects
What is easier than expected?

- Overlapping Test phases
- End User Adoption
- Use of off-shored test resources
  - Functional
  - Performance
Enhancements to testing methodology for cloud testing

- Test planning
- Test design
- Cloud setup
  - Requires familiarity of the target Cloud environment and of its key functions
  - Understanding of Cloud monitoring tools is essential
- Test execution
  - Requires usage of additional resources (computational units, storage, communication channels), possibly TaaS
- Remediation
- Closure
  - All used resources need to be released
  - Storage resources could be used for archiving test cases, results, used VMs could be saved as images and for later use
Testing focuses when migrating an application in the Cloud

- Test application changes needed to adapt the application to the Cloud
- Check on-premise interfaces
- Check data migration and backup mechanisms
- Focus on new security aspects that may have added
  - Encrypted data in cloud storage
  - New secure channels
- Performance and scalability
  - Compare with on-premise setting
  - Make sure that you are making good use of the features offered by the Cloud
- Availability and disaster recovery
CLOUDBEES
CloudBees main characteristics (1)

- Differently from GAE (and Azure), relies on well-known, state of the art components
  - Tomcat
  - Jboss 6 Application Server
  - MySQL
- Can execute any standard Java web application
- An application built for CloudBees can be moved to any other environment, provided that you install the state of the art components it uses
  - No vendor lock-in
- Offers a load balancer and manages an application cluster
  - The developer, however, is in charge of maintaining HTTP session on some repository
CloudBees main characteristics (2)

- Supports a continuous integration process through an on-demand Jenkins (Hudson) service
  - Each time a new version of a software component is released, the system is automatically built and verified against an array of tests
  - The build process can take into account the definition of dependences in a Maven repository
Examples

- New Relic: performance monitoring service
- Papertrail: aggregation and analysis of log files
- Sauce Labs (Selenium): automated in-browsed testing of web applications
- MongoHQ (MongoDB): noSql database
- SendGrid: SMTP service
References

- Cloud testing
  - Parveen, T.; Tilley, S.; , "When to Migrate Software Testing to the Cloud?," Software Testing, Verification, and Validation Workshops (ICSTW), pp.424-427, 6-10 April 2010
  - SOASTA website http://www.soasta.com/
References

- Testing cloud-based applications
  - RAJAGOPAL SATTALURI, Testing Considerations for Application Migration to Cloud Computing, [http://cloudcomputing.sys-con.com/node/1703085](http://cloudcomputing.sys-con.com/node/1703085)

- CloudBees