Whitepaper

operational acceptance testing

Business continuity assurance

Trusted partner for your **Digital Journey**

Atos

Table of contents

For Reference please contact:

Sriini Vaasudevan
Test Lead & Operations Manager/
Product Owner – Telco CRM
Eemsgolaan 7, 9727 DW Gron-
ingen, Netherlands
+31 (0)683624587
sriinivaasan.vaasudevan@atos.net

Table of figures

Summary

Operational Acceptance Testing (OAT) is the penultimate phase of Acceptance Testing in a Software Testing Life cycle and is the last defence line between a software development project and deployment of software on production. FFor decedes, Operational Acceptance has been undermined and misunderstood. Where User Acceptance has been written about and hailed as a "final phase in testing before production". User Acceptance is but one side of the coin, Operational Acceptance is the other.

The International Software Testing Qualifications Board (ISTQB) defines 'Acceptance Testing' and 'OAT' as:

Acceptance Testing
Formal testing with respect to user needs, requirements, and business processes conducted to determine whether or not a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system.

Operational Acceptance Testing
Operational testing in the acceptance test phase, typically performed in a (simulated) operational environment by operations and/or systems administration staff focusing on operational aspects, E.g. recoverability, resource-behavior, installability and technical compliance.

Irrespective of the delivery model employed for the project (waterfall, agile, devops), 3 aspects need to be addressed:

1. Conformance of the asset built with the requirements and specifications
2. Operate and support the asset once deployed to production
3. Ensure adherence with Business KPI, IT SLA and business continuity.

To address these questions, the project team needs to perform the Operational aspects of acceptance testing and evaluation commonly known as Operational Acceptance Testing (OAT). This whitepaper evaluates the quality characteristics associated with operational acceptance test scope from the perspective of the newly released software testing standard ISO 29119.

Introduction

Initially it used to be a practice for testing to be part of the Software Development Lifecycle (SDLC) wherein the development teams did a test analysis on their own product leading to impartially within their own team. Through experience and growing sophistication in the software, software units evolved multiple phases for testing to improve the quality of the end products produced. During the 1980's Paul Rook designed the V-Model methodology to improve the overall efficiency and effectiveness in software development processes. By this time, the differing test phases had grown to include: Unit Testing, Component Testing, System Integration Testing (SIT), System Test (ST) and Acceptance Testing.

References to 'acceptance testing' were (then) understood and interpreted as to mean business or User Acceptance Testing (UAT). As a result of this interpretation, UAT was often perceived to be the last, or one of the last, lines of defense between a software development and its implementation into the production environment. But this misconception started backfiring. Hereby below a SOC.

In a project, when the project teams completed the development of the software, it was released and handed over to the operation team and the application owner. It immediately became a part of the business processes as it was launched into the production environments. Consequently, known and unknown defects of the software will directly impact business continuity and potentially cause damage to a greater or lesser extent. In addition, responsibility typically is transferred from the development units to two stakeholders:

Application Owner

The single point of contact for business units concerning the operation of dedicated applications. These units are the internal part of line organization managing the maintenance, and constitute the interface to operation teams.

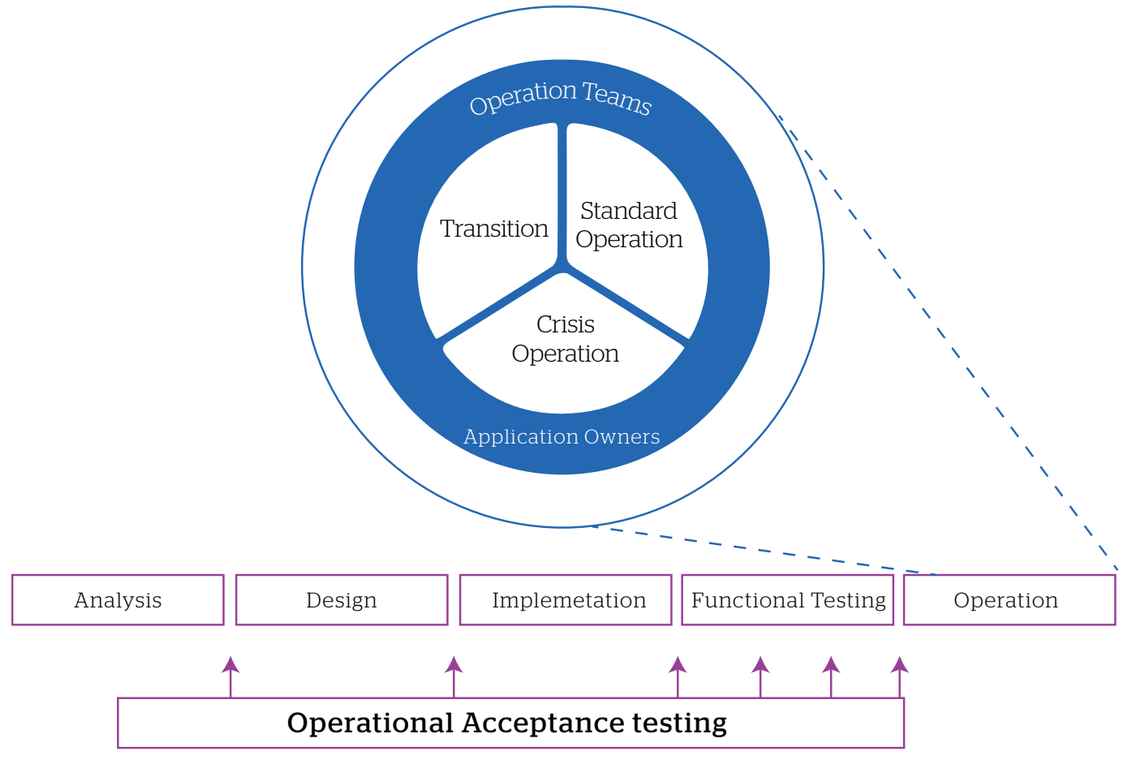
Operation Team

An internal or external team deploying and operating software following well-defined processes, e.g. tailored by Information, Technology, Infrastructure, Library. These units are equipped with system and application utilities for managing the operation, they will contact the application owners if bigger changes should be necessary to guarantee business continuity.

This evolution in Information, Technology, Infrastructure, Library provoked the fact that Business continuity cannot be only assured by purely functional quality assurance, this created a need for introducing various non functional aspects borne out of the ITIL processes (Information Technology Infrastructure Library). This new paradigm signaled a clear evolution from a purely functional scope to a more holistic scope of acceptance testing – encompassing both functional and non-functional aspects to ensure business continuity. After this new paradigm of acceptance had been accepted by the development and testing communities, a new understanding Operational Acceptance Testing (OAT) came on board. By 2010 the software quality industry published the ISO 25000 SQuaRE [REF-5] series of standards that outlined the scope of OAT, in respect to the evaluation of quality requirements, at a framework and strategic level. In 2013 the Testing industry published the ISO 29119 Software Testing series of standards, which enabled effective and efficient testing by qualified/quantified means to support the delivery of reliable, robust IT assets.

The present paper gives an overview of how to use ISO 25000 (ISO/IEC JTC 1/SC 7 Software and Systems Engineering, 2010) to scope out OAT systematically and how to apply test methods successfully by having application owners and operation teams involve other stakeholders like architects, developers, or infrastructures.

Figure 1 : Main Activities and OAT stakeholders



Scope of Operational Acceptance Testing

In traditional sequential development method, the Operational Acceptance testing (OAT) is likely to be executed near the end of the software development life cycle. This would enable the test to be executed on a like or near-production instance of the asset that is to be implemented. This contributes to the idea of Operational Readiness and Assurance. Let's define them :

- **Operational Readiness** is the process of preparing the future asset owner, and the support team, so that, at the time of implementation/cutover, they are fully ready to assume ownership and operation of the asset.
- **Assurance**, in this context, refers to the act of re-assuring the project and organization stakeholders that their (iterative) developing asset, and their organization, is in a state of operational readiness – or by which to provide a measure of assurance that the asset will be ready in the adhered time frame.

OAT subsumes all test activities performed by application owners and operation teams to arrive at the acceptance decision to operate the software under agreed Service Level Agreements and Operation Level Agreements (SLAs / OLAs).

These agreements provide measurable criteria which - if they are fulfilled - implicitly ensure business continuity.
Operation capability covers three groups of tasks:

- **Transition:** This designates the successful deployment of software into the production environment using existing system management functionality. The production environment could be a single server or thousands of workstations in agencies. Software deployment includes software, database updates, and reorganization but also fallback mechanisms in case of failures. The operation team has to be trained and equipped with the respective tools.
- **Standard Operation:** This designates successful business execution and monitoring on both the system and the application level. It includes user support (helpdesk for user authorization and incident management) and operation in case of non-critical faults, E.g. switching to a mirror system in case of failure analysis on the main system.
- **Crisis Operation:** System instability causes failures and downtimes. Operation teams must be able to reset the system and its components into defined states and continue stable standard operation. Downtimes have to be as short as possible, and reset must be achievable without data or transaction loss and/or damage.

Delivery Models

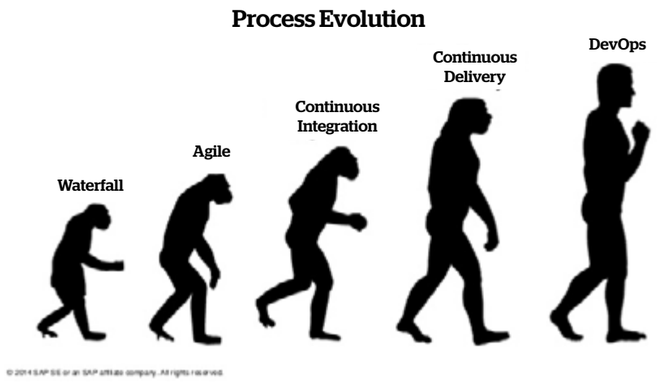
The essentiality of OAT can be achieved through the process of incorporating it in the delivery models. In the growing evolution of software development, the delivery models have matured. From the primitive Waterfall to Agile which is rationalized now in the growing rage to go Devops.
Below a definition scale of all 3 and if OAT is coherent with them.

Waterfall or V Model - The waterfall model is a sequential design process, used in software development processes, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of conception, initiation, analysis, design, construction, testing, production/implementation and maintenance.

Agile Model - Agile development model is also an incremental model. Software is developed in incremental, rapid cycles. This results in small incremental releases with each release building on previous functionality. Each release is thoroughly tested to ensure software quality is maintained. It is used for time critical applications.

Dev Ops - A combination of Development & Operations – is a software development methodology which looks to integrate all the software development functions from development to operations within the same cycle.

Figure 2: Software Model Process Evolution



The relevance of the different types of OAT is derived from the individual artefacts and their quality attributes. The major test types are allocated, and decisions about test execution follow from the risk evaluation - the scope of OAT is ultimately defined by selecting columns from the table shown in Figure 3 and substantiation of these types in coherence with the delivery models is explained below as part of section 3.1.

Figure 2 : Quality attribute evaluation

	Operational Documentation Review	Code Analysis	Rehearsal Testing	Installation Testing	Framework/ Platform Upgrade Testing	SLA/OLA Monitoring test	Load/ Performance Testing	Security testing	Backup & Restore Testing	Failover Testing	Recovery Testing
Transition	X	X	X	X	X						
Standard	X				X	X	X	X			
Crisis	X				X	X	X	X	X	X	X
Functional Suitability	X		X	X	X	X	X	X	X	X	X
Performance Efficiency		X		X	X	X	X		X		
Compatability				X	X						
Usability	X		X				X				
Reliability	X		X			X			X	X	X
Security	X			X		X		X			
Maintainability	X	X	X			X	X			X	
Portability	X		X	X	X						

Types of Operational Acceptance Testing

Operational Documentation Review

Definition / Objective	All Documents that are necessary to operate the system. (e.g. architecture overviews, design documentation, operational docs) are identified, and they are checked for completeness, availability and accessibility to the relevant people. Transition, Standard and Crisis mode are addressed.
Input	All prerequisites to implementation. Few examples are: » IBR (Infrastructure Blue prints) » Implementation Plan » Installation Manual and Application Production Manual » Architecture Overview » Business Continuity/Disaster recovery plans
Test Approach	The task of the OAT tester is to ensure that these documents are available finalized and in sufficient quality to enable smooth operation. The documents must be handed over and accepted by the teams handling the production support, helpdesk, disaster recovery, and business continuity. The handover must be documented. Operation teams must be included as early as possible to obtain their valuable input about the documentation required for operating the system. This way, transparency about the completeness of documents is achieved early on in the development process, and there will be sufficient time left for countermeasures if documents are missing or lacking in quality.
Delivery Model	**Waterfall Model -** This type of OAT is specifically designed for Waterfall Model. There is a higher cost involved in following the overall handover to operations. **Agile or Devops Model -** This type of OAT is usually not followed rigorously in this model. But this can be easily covered by the close cooperation in the scrum teams wherein project, business and operations are involved. The costs involved are less as no proper handover is needed in this approach.
Risk of Not Executing	If the operational documentation review is omitted or performed to late, testing will start without the final documentation available which reduces the effectiveness of the tests. As a result, an increased number of issues may be raised, causing delays in the timeline. For operation teams, not having the correct documentation can affect their ability to maintain, support and recover the systems.

Code Analysis

Definition/ Objective	Code Analysis is a tool based automated and/or manual analysis of the source code with respect to dedicated metrics and quality indicators. This type of test is to ensure transparency about the maintainability and stability of software. Applying correction and refactoring increases handling of incidents needing hot fixes with little effort in terms of regression testing as possible.
Input	Source code of implemented software. Coding and architecture guidelines.
Test Approach	The task of the OAT tester is to do the following: » Selecting the relevant code and tool set-up » Executing code analysis » Assessing results and thresholds concerning quality attributes » Analyzing trends with respect to former analysis » Performing benchmarking with respect to standards » Deciding on acceptance or rejection
Delivery Model	**Waterfall Model -** This type of OAT is handled as part of unit testing in the waterfall model. Fully coherent and necessary phase. **Agile or Devops Model -** This type of OAT as a principle is also carried forward to Agile or Devops model. Static analysis is a quality technique where an automated tool checks for defects in the code, often looking for types of problems such as security defects or coding style issues (to name a few). An accelerator developed within Atos for this purpose is CAST tooling is used for code inspection, assigning mixed teams to increase the bandwidth at transition points (e.g. developers involved in testing, testers with operations go-live), agile, lean and integrated methods (devops, agile testing), continuous innovation and transformation of the IT and process landscape, systematic capture of knowledge and systematic reuse of solutions across domains.
Risk of Not Executing	High risk of side effects besides functionality defects Greater effort required in regression and Rehearsal testing

Rehearsal testing

Definition / Objective	Unlike User Acceptance testing / E2E which integrates business functions. OAT integrates all functions and stakeholders of a production system. A rehearsal or staging environment testing is necessary for bigger changes in production environments, especially when they concern many stakeholders. Objective is to avoid the risks of failures in the process chain and longer system downtimes shall be minimized or avoided.
Input	All prerequisites to implementation. Few examples are: » Fully tested software with the test release recommendation implementation plan » Installation Manual and Application Production Manual » Business Continuity / Disaster recovery plans.
Test Approach	» The main fundamentals in this phase are to ascertain that the test cases of functional change when executed ensure business continuity. » 2 sets are needed, Roll-forward and Roll-back scenarios. » The test team in close conglomeration with operations has to execute the relevant test cases that adhere to various factors based on the project being implemented. » In case of Business projects, new functionality introduced is the main focus. » In case of LCM projects, all major modules including technical behavior of the system needs to be scrutinized.
Delivery Model	**Waterfall Model -** This type of OAT is carried out rigorously to make sure that the operational requirements are handled. This is usually difficult in waterfall as the collaboration with operations is not so strong yielding to slower time to market and higher costs. **Agile or Devops Model -** This type of OAT is seamlessly integrated in the agile/devops methodology due to the involvement of project, business and operations in the same team. Due to this integration, it is more effectively executed with faster time to market and reduced costs in this model.
Risk of Not Executing	Implementation itself is at risk Business continuity at risk IT adherence of SLA is at risk Major leakage of incidents is also possible.

Installation Testing

Definition/ Objective	This test activity ensures that the application installs and de-installs correctly on all intended platforms and configurations. Objective is to ensure correctness, completeness and successful integration into system management functionality for following : Installation, De-installation, Fallback, Upgrade, Patch.
Input	Target systems with different configurations (e.g. created from a base image), as well as the specification of changes to the operating system (e.g. registry entries), and a list of additional packages that need to be installed.
Test Approach	The following aspects must be checked to ensure correct installation and de-installation: » The specified registry entries and number of files available need to be verified after the installation is finished. » The application must use disk space as specified in the documentation in order to avoid problems with insufficient space on the hard disk. » If applicable, the installation over old(er) version(s) must be tested as well the installer must correctly detect and remove or update old resources. » The occurrence of installation breaks has to be tested in each installer step, as well as breaks due to other system events (e.g. network failure, insufficient resources). The application and the operating system must be in a defined and consistent state after every installation break possible. » Shared resources may have to be installed or updated during installation, and while these processes are performed, conflicts with other applications must be avoided. In terms of uninstallation, the system should be cleaned from shared resources that are not used any more. This will result in higher performance and increased security for the whole system. » Since the installation routine is an integral part of the application and a potentially complicated software process, it is subject to the regulations of ISO 25010.
Delivery Model	**Waterfall Model -** This type of OAT is poorly handled due to the handing over process from a previous phase to the next. Each phase is executed with a set team. Nevertheless it is part of the process followed in this model. **Agile or Devops Model -** This type of OAT is usually well covered in this model due to the team collaboration and clarity of requirements. Continuous deployment and testing are highly adhered as part of this type of testing.
Risk of Not Executing	May result in conflicts with other applications Compromise or even broken systems Low user acceptance even-though the software itself has been tested successfully and works as designed. Number of manual effort, cost and conflicts in larger systems

Framework / Platform Upgrade Testing

Definition / Objective	This type of testing comprises test activities that ensure successful exchange or upgrade of central components like run time environments, database systems or standard software versions. Objective is to obtain proof of correct functionality, sufficient performance or fulfillment or other quality requirements.
Input	IBR (Infrastructure Blue prints) Functionality in chain that needs regression testing
Test Approach	2 possible approaches for introducing central components: » The first approach would be to set up central components as productive within the development system, i.e. central components would move parallel to the application software along the test stages towards a release date according to a common release plan. Testing would start implicitly with developer tests. » The second approach would be to test changing a central component in a production-like maintenance landscape. In this case, a dedicated regression test would be performed parallel to production. Central components would be released for both operation and development. » Based on the approach the steps would be : 1. Deriving relevant applications from impact analysis 2. Selecting regression tests on the basis of risk assessment 3. Performing regression tests (including job processing) – Parallel to development – In a dedicated maintenance environment 4. Deciding on acceptance or rejection
Delivery Model	**Waterfall Model -** This type of OAT is followed in this model. The main drawback is the integration across teams. Usually this type of testing needs test strategy to cover the various requirements to be testing in the old and new platform. Often this is a slow process in waterfall methodology. **Agile or Devops Model -** This type of OAT is followed in this model. The main advantage is the integration across teams. Usually this type of testing needs test strategy to cover the various requirements to be testing in the old and new platform. At times multiple scrum teams can be setup to cover the requirement of testing across platforms.
Risk of Not Executing	Incompatible software platform System downtimes Non-functional issues affecting business continuity Missing fallbacks Data defects Very crucial for Multi-vendor environment and cloud computing

SLA / OLA Monitoring Testing

Definition / Objective	This test type examines the implemented monitoring functionality in order to measure the service and operation level. Objective is to derive if the monitoring functionality is complete, correct and operable in order to derive the right service and operation level.
Input	Business Continuity Checks KPI parameters Thresholds and warning requirements
Test Approach	Select relevant SLA/OLA Deriving Monitoring scenarios to estimate service levels Integrating scenarios into test scenarios of other test types Execute tests and calculating service levels from monitoring
Delivery Model	**Waterfall Model -** This type of OAT is not specifically designed for Waterfall Model. There is a higher Cost involved in following the overall handover to Operations. **Agile or Devops Model -** This type of OAT is usually followed rigorously in this model. But this can be easily covered by the close cooperation in the scrum teams wherein project, business and operations are involved. The costs involved are less as no proper handover is needed in this approach.

Load / Performance Testing

Definition / Objective	Performance testing is a technique used to ascertain the parameters of the asset in terms of responsiveness, effectiveness and stability under various workloads. This process involves quantitative tests performed to measure and report on the load, stress, endurance, volume and capacity threshold limits of the asset. Performance testing measures the quality attributes of the system, such as scalability, capacity and resource utilization.
Input	All prerequisites to implementation. Few examples are: IBR (Infrastructure Blue prints) Architecture Overview Application production manual highlighting the measures. Requirements for Performance and Load
Test Approach	» Collecting test requirements from an operational point of view » Integrating requirements into the Performance/Load model » Preparing the test environment and test data » Executing and analyzing the test » Defining mitigation scenarios for performance risks » Deciding on acceptance or rejection
Delivery Model	**Waterfall Model -** Development life cycle times in waterfall model are slow. Performance testing is usually started very late in this life cycle and the test engineers face the pressure of finishing on time. Another performance related issue is reliably using data from a limited nonproduction environment to precisely predict how the system can perform in a more robust environment. **Agile or Devops Model -** Agile development breaks through the barriers of traditional waterfall model to software development to provide better value faster and improve the ROI (return on investment). Performance testing that usually takes place at the end of the lifecycle in a waterfall model will move to the beginning in an Agile methodology, including the analysis and designing. The performance of an application is directly proportional to its design and hence should be addressed at an early stage of the development lifecycle. Performance testing is considered throughout the Agile process that is from the release planning stage onwards.
Risk of Not Executing	Performance issues many go unnoticed after going live Business continuity interruption SLA/OLA adherence hampered

Security Testing

Definition / Objective	ISO defines this as a "type of testing conducted to evaluate the degree to which an asset, and associated data and information, are protected so that unauthorized person or systems cannot use, read or modify them, and authorized persons or systems are not denied access to them."[REF-4] It is a technique used to ascertain if the asset protects the data and maintains the functionalities as intended; in respect to authentication, authorization, availability, confidentiality, integrity and non-repudiation.
Input	Risk and Vulnerability Analysis document Test documentation Configuration files
Test Approach	For all relevant input channels: » Check that trash data is handled carefully (fuzzing data). i.e. ensure that the CIA concept (Confidentiality, Integrity, Availability) is still met » Check that the overall system works properly when system components are switched off (e.g. firewall sniffer. proxies). For all components: » Check that all test-motivated bypasses (e.g. test users, test data) are deleted for all configuration items » Check that all sensitive data are handled according to security guidelines » Check that no debugging information is shown and error messages are customized (information disclosure) » Check that standard credentials are changed to individual and secure credentials » Check the configuration, switch off unused components and close unused ports (firewall) » Check used certificates for validity.
Delivery Model	**Waterfall Model -** During the testing phase of the Waterfall Model, the QC team have a full set of functional requirements to test. These requirements are to be gathered in the requirements specification phase. Assuming that security was integrated into the previous phases of the SDLC, the QC team will also have a set of security requirements to validate. Security requirements go through the same creative process as normal functional requirements. Disadvantages are findings from early security reviews are often ignored as "theoretical" and costly to go backwards in the development timeline. **Agile or Devops Model -** This type of OAT is usually followed rigorously in this model. But this can be easily covered by the close cooperation in the Scrum teams wherein Project, Business and Operations are involved. The Costs involved are less as no proper handover is needed in this approach. Advantage of integrating security testing in this model are: » Unit tests include security mechanisms & integrate peer code reviews » Test input validation by verifying behavior in edge cases » Test access control by verifying behavior from multiple roles.
Risk of Not Executing	Security vulnerabilities Privacy issues resulting leakage of customer data System/ server hack

Backup & Restore Testing

Definition / Objective	Backup and restore testing focuses on the quality of the implemented backup and restore strategy. This strategy may not only have an impact on the requirements for software development but also on SUs that have to be fulfilled in operation. In an expanded test execution, the test objective of a backup includes all the resources, ranging from hardware to software and documentation, people and processes.
Input	Working test environment and operation process
Test Approach	Backup and restore testing can be executed in a use-case scenario based on well-defined test data and test environments. In general, a test will comprise the following steps: » Quantifying or setting up the initial well-defined testing artefacts » Backing up existing testing artefacts » Deleting the original artefacts » Restoring artefacts from backup » Comparing original artefacts with restored ones. » If applicable, performing a roll-forward and checking again.
Delivery Model	**Waterfall Model -** This type of OAT is not specifically designed for Waterfall Model. There is a higher cost involved in following the overall handover to operations. **Agile or Devops Model -** This type of OAT is usually followed rigorously in this model. But this can be easily covered by the close cooperation in the scrum teams wherein project, business and operations are involved. The costs involved are less as no proper handover is needed in this approach.
OAT Test Environment	If backup and restore functionality is available, testing can in principle be executed parallel to early functional testing. However, since the tests will involve planned downtimes or phases of exclusive usage of environments. Moreover, this activity will require the following: » Representative test data » Established backup infrastructure » Established restore infrastructure.
Risk of Not Executing	Risk of losing data in a restore situation impeding ability to perform disaster recovery Business continuity interruption SLA/OLA adherence hampered

Failover Testing / Recovery Testing

Definition / Objective	The objective of failover testing can be subdivided into two categories: » The degree of the quality of fault recognition (technical measures have to be implemented to detect the failure event e.g. a heartbeat) » The efficiency and effectiveness of the automatic failover reaction in terms of reaction time and data loss.
Input	IBR (Infrastructure Blue prints) Installation Manual and Application Production Manual Architecture Overview/ Failover plans
Test Approach	The test case specification has to describe the measures taken to trigger the failure event. It is not necessary to execute events exactly as they happen in the real world since it can be sufficient to simulate them with technical equipment. For instance: » Failure: Lost Network Connection » Failure: File system or hard disk failure.
Delivery Model	**Waterfall Model -** Development life cycle times in waterfall model are slow. Failover testing is usually started very late in this life cycle and the test engineers face the pressure of finishing on getting accurate results from performance testing in time. The major disadvantages foreseen in this model are late start in the lifecycle, unclarity of requirements. **Agile or Devops Model -** Agile development breaks through the barriers of traditional waterfall model to software development to provide better value faster and improve the ROI (return on investment). The failover and recovery of an application is addressed at an early stage of the development lifecycle.

Facets of Operational Acceptance Testing

The objective of OAT is to achieve the commitment of a handover of the software to the application owner and the operation team. Acceptance is based on successful tests and known but manageable defects throughout the entire life cycle of the software development project. The testing is organized among all the stakeholders.
The application owner and the operation team

- perform a test of their own which is integrated into the overall test plan
- support the testing teams in achieving the test results
- check third-party results with respect to acceptance criteria.

Consequently, operational acceptance starts early on in the project and is achieved after successfully passing all the quality gates.

Efficiency and effectiveness

Efficiency in Operational Acceptance testing needs to be assured and some methodologies that can serve this purpose are listed as below

1. Early defect detection with Business Process Validation (BPV) and Requirements Validation
2. Risk based testing approach
3. Traceability of testing requirements
4. Regular Test Assurance for accuracy and quality of test cases and executions
5. Test Automation including Model-based testing techniques for increased quality and efficiency
6. Consultant/ Application owner and Operations led test health check
7. Service Virtualization

The other aspect of efficiency needing continuous assurance of quality is effectiveness. For effectiveness, a set of quality gates have to be defined which collect all the information about the various operation tests and allow decisions to be made concerning acceptance. Below you can see the implementation of the efficiency principles as gates in the life cycle.

Figure 3 : Main test phases of OAT Types

Conclusion

Current situation & outlook

Operational Acceptance Testing (OAT) is a crucial part of the Software Development Life-Cycle (SDLC). OAT is where the underlying software configurations and operational support components come together. This process tests the implementation of structural or functional changes to a software or service within a functional or non-functional environment. In other words, Operational Acceptance Testing evaluates whether or not an application can be deployed to a network according to IT Infrastructure Library (ITIL) standards. OAT determines if a software will operate the way it is designed to without disrupting the whole installation, network or business that uses it.

Ultimately, OAT should focus on the resiliency, recover ability, integrity, manageability and supportability of a software or network installation. There should also be separate testing processes for performance, security and data loss/disaster recovery, which are, themselves, specialty areas of huge importance in their own right.

Change Driven Risk Management (CDRM) is the process that determines exactly how much Operational Acceptance Testing is needed. This CDRM process will lead to the appropriate risk assessment strategy for a new installation project. As a result, the OAT process will be more efficient and focused in identifying and addressing operational risks.

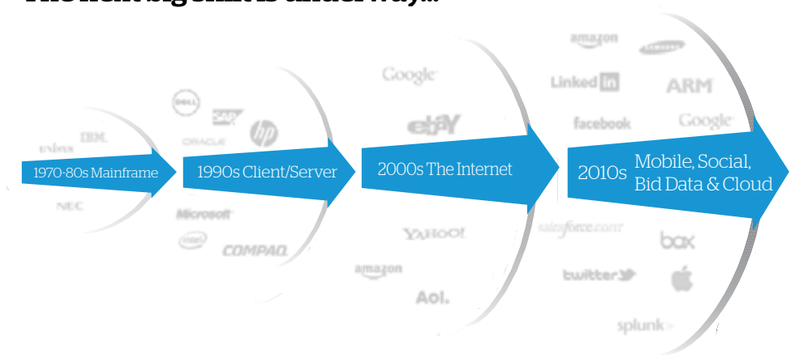
The next big shift – Automation

Apart from the general requirements outlined above, companies are also be influenced by the implementation models used by cloud service providers. Gartner recommends that enterprises should apply the concepts of cloud computing to future datacenter and infrastructure investments in order to increase agility and efficiency. This, of course, affects operational models, and OAT will have to support business continuity in this growing market. Forrester estimates that cloud revenue will grow from $41 billion in 2011 to $241 billion in 2020 (Valentino-DeVries, 2011). Today, OAT is already being executed by a varíety of stakeholders. Target groups are specialists like application owners or operation units in large organizations, as well as individuals in the private sector. Internal departments and / or vendors need to be addressed to establish sustainable processes. OAT should follow a systematic approach so as to mitigate the risks of cloud providers and outsourcing partners, because companies which offer services using these types of sourcing will not notice any incidents but their impact will be felt directly by the clients.

In this age of instant gratification and ubiquitous personal technology, changes to software or a network are expected to be implemented seamlessly. End-users now expect applications and services to be fully functional upon release. Users also expect a steady stream of software updates to be implemented in a way that do not hinder the overall utility of the tool. Hence, automating the OAT process is becoming increasingly unavoidable.

Developers have traditionally tried to address this by writing run-time tests as they code, essentially implementing manual unit testing for infrastructure while building an application. While this can be effective, it adds additional time to the original development cycle, which undoubtedly leads to cost increases.

Figure 4 : The future of Operational Acceptance Testing

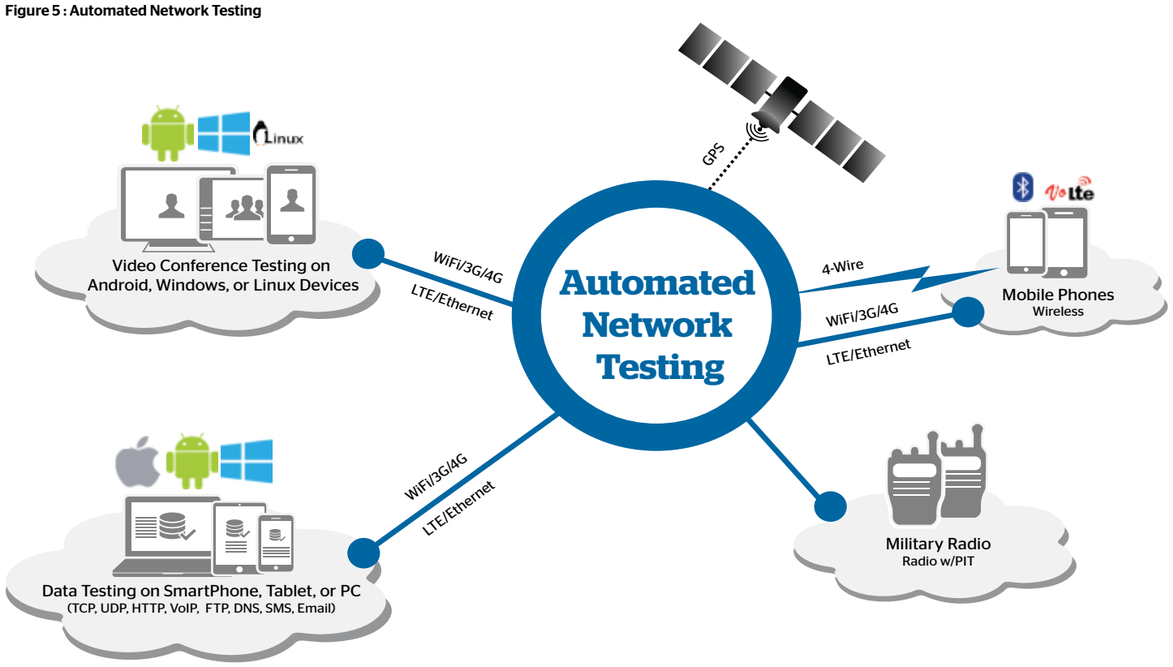


Some of the research done in this area has led to the following evolutions and standards to cope with the above mentioned requirements.

The efficiency of an automated network testing tool can shrink testing and recovery time from weeks to days - or even hours, if using the right system. Automating operational testing also can save revenue in the long-term, by reducing both cost of development and cost of maintenance. To that end, automating OAT has been somewhat of a holy grail amongst developers, enterprises and network administrators.

Thankfully, several tools have made their way to the market that offer varying levels of testing and management automation. All three offer detailed reporting of testing results, making ALM (Application Life-cycle Management) easier for administrators. These are highlighted below :

Figure 5 : Automated Network Testing



The Bridge Solution – Atos

With its overall business process based view on the landscape, the Atos Bridge provides the operational single-point of contact towards Atos' customers. The unique combination of skills and capabilities for this centralized solution, one Operator provides Application, System and Database management, gives the most efficient and effective support possible. On the Atos Bridge the landscape is monitored 24x7 and any unexpected events are handled, regardless of the delivery party responsible or the technology being used. The Bridge provides the following services:

- Business Process-oriented Monitoring
- Single point of contact (SPOC)
- ISI (infrastructure Service Integration) coordination towards 3rd parties
- Application Management
- System Management
- Security Management
- Database Management
- Backup and Restore services
- Daily operations and Service Work Request execution
- Incident, Problem and Change management (SMC on Bridge)
- Reporting services

Figure 6 : Atos Bridge & its services

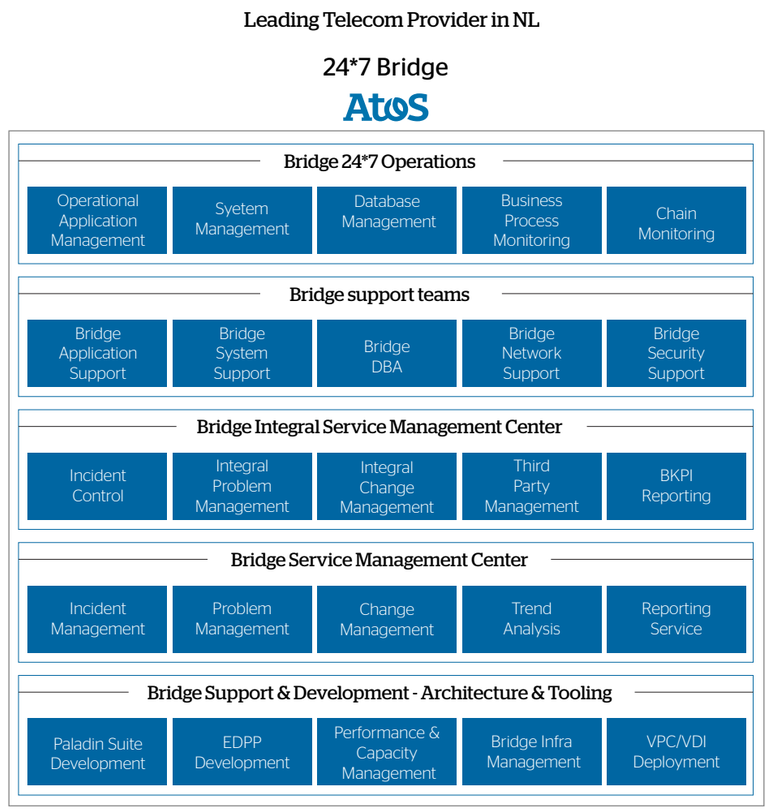


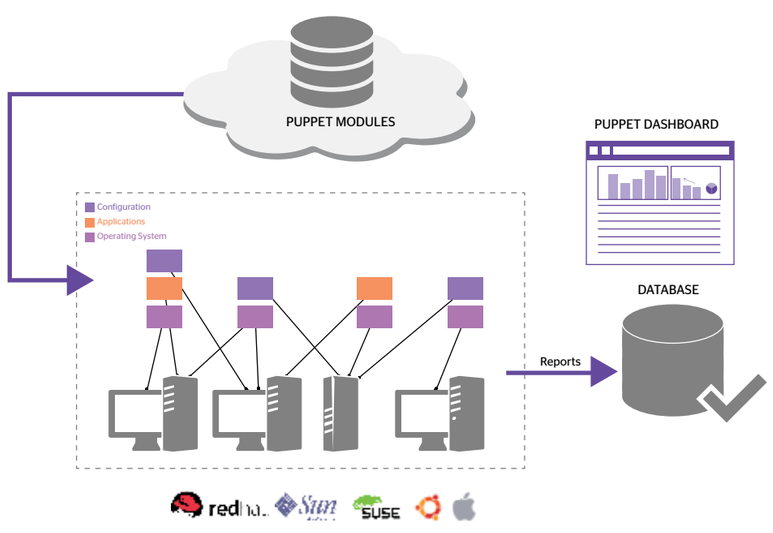
Figure 7 : Paladin – Business Process Oriented Monitoring

Opscode's Chef is an open-source systems framework tool specifically designed for automating cloud-based services. The framework offers users a fully automated infrastructure. Users write abstract definitions, which are then used as source code to build unique parts of the infrastructure. These definitions are then applied to individual servers. The tool is a Configuration Management System (CMS) built upon a system integration framework, allowing step-by-step automated testing for your network. The over-all functionality of the tool can even be expanded using downloaded plug-ins, known as Cookbooks. As an open-source framework, it is easily expandable and adaptable for and by the community of users. Opscode's software runs a native version of the Ruby on Rails programming language, which means that users must be familiar with the code and its syntax requirements.

Figure 8 : Opscode Chef

Puppet for Enterprise software is an IT automation software that gives system administrators the power to easily automate repetitive tasks (including Operational Acceptance Testing). The tool also quickly deploys critical applications and proactively manages infrastructure changes, whether executed on-premise or remotely from the cloud. Any task along the way of managing your network can be automated using the application. A model-based approach automates repetitive tasks and simply eliminates configuration drift. The software enforces the desired state of your infrastructure that you define, even as you work on other projects within the application. The application uses its own unique coding language, which will require users to be familiar with its syntax. Fortunately, this has been designed to be as user-friendly as possible.

Figure 9 : Puppet Labs



References & Annexes

1. APM Group Limited, HM Government and TSO. ITIL - Information Technology Infrastructure Library. High Wycombe, Buckinghamshire. [Online] 2012. http://www.itil-officialsite.com.
2. International Software Testing Qualifications Board. ISTOB Glossary. [Online] March 2010. http://www.istqb.org/downloads/glossary.html.
3. ISO/IEC JTC 1/SC 7 Software and Systems Engineering. ISO/IEC 25000 Software Engineering - Software Product Quality Requirements and Evaluation (SQuaRE) - Guide to SQuaRE. [Online] 17/1212010. http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_deta il.htm?csnumber=35683.
4. Software Engineering Institute. Architecture Tradeoff Analysis Method. Carnegie Mellon University, Pittsburgh, PA. [Cited: 2012.] http://www.sei.cmu.edu/architecture/tools/evaluate/atam.cfm.
5. Wikipedia contributors. FMEA - Failure Mode and Effects Analysis. [Online] 2012. http//de.wikipedia.org/wiki/FMEA.
6. Operational Acceptance Testing – ScriptRock https://www.scriptrock.com/
7. Atos MS Solutions - The Atos Bridge 1.4.pdf

About Atos

Atos SE (Societas Europaea) is a leader in digital services with pro forma annual revenue of circa € 12 billion and circa 100,000 employees in 72 countries. Serving a global client base, the Group provides Consulting & Systems Integration services, Managed Services & BPO, Cloud operations, Big Data & Cyber-security solutions, as well as transactional services through Worldline, the European leader in the payments and transactional services industry. With its deep technology expertise and industry knowledge, the Group works with clients across different business sectors: Defense, Financial Services, Health, Manufacturing, Media, Utilities, Public sector, Retail, Telecommunications, and Transportation.

Atos is focused on business technology that powers progress and helps organizations to create their firm of the future. The Group is the Worldwide Information Technology Partner for the Olympic & Paralympic Games and is listed on the Euronext Paris market. Atos operates under the brands Atos, Atos Consulting, Atos Worldgrid, Bull, Canopy, Unify and Worldline. Management