

## Automation Solution for Installation Testing of Document Processing and Delivery Software

### Customer

The customer provides top notch solutions for document processing and delivery.

<b>Company</b>	<i>Software Development Company</i>
<b>Country</b>	<i>United Kingdom</i>
<b>Business Domain</b>	<i>Document Management Software</i>
<b>Services Used</b>	<i>Test Automation</i>
<b>Cooperation Model</b>	<i>QA Outsourcing for a Development Company</i>
<b>Duration</b>	<i>6 months</i>
<b>Efforts</b>	<i>2 man-months</i>

### Project

The project automates a cross-platform solution for content processing and delivery, designed for corporate clients. The idea behind this technology is that the central control unit sends faxes and SMS-messages from users' workstations through a specially allotted server. One of the product's specific features is integration with third-party document processing and managing solutions. The product enables administrators to minimize delivery expenses by granting privileges for the dispatch and receipt of faxes and SMS-messages.

### Challenge

The QA team was assigned the task of automating installation testing of the product's constituent parts – a client and server. The assignment included a number of mandatory requirements:

- Install the client and server on Windows-based operating systems.
- Configure the basic settings of the server and client.
- Perform the initial check of the application operability.

### Solution

The automation team selected TestComplete for the project automation because this tool provides a number of benefits:

- TestComplete is a lightweight test executor.
- TestComplete supports distributed testing, that is, it allows running multiple tests on several machines simultaneously, and during the test run test parts interact with each other.

The automation experts prepared 5 test environments (5 virtual machines) on different Windows versions and developed 6 basic scripts for installation and setup of the application under test. The team also created a TestComplete shell in Ruby to run continuous tests and control test results. As a result, a typical automated test run followed this scenario:

- Copy a clean pre-configured virtual machine (VM) into a temporary folder.
- Launch an operating system on the VM.
- Launch a remote test script on the VM.
- Retrieve test logs.
- Turn off the VM and remove it from the temporary folder.

## Technologies used

**Test tool:** TestComplete

**Programming languages:** JavaScript, Ruby

**Other tools:** VMware

## Success

The automation team developed a suite of automated tests that covered the basic functionality of the application running on various operating systems. In previous project stages of the development cycle, these tests were performed only once. Using the automation solution, the customer can test each product build.

The duration of a single test run is 24 hours. Analogous manual test activities require almost 40 hours, i.e., one working week. As a result, the customer receives detailed reports on the application compatibility within a day. Now the development team can promptly fix uncovered defects and make any other associated modifications, which is the essential benefit of early defect detection.