iT People Corporation

Automation Testing – Case Study
Our Client and the Project Context

The client is a leading insurance company that provides health insurance benefits to millions of its clients.

The project was to automate the tests originally developed for a legacy claims-processing system on a newly built rich web interface. While most functionality remains similar, certain flow-level changes have been made in the reengineering of the application and manual testing of such a system is prohibitively expensive. IT People has been taken to achieve optimum level of Test Automation, and has been successful in doing so, within the agreed time and budget.

The scope of test automation primarily included functional test areas, and capacity / performance testing of the critical usage areas.

Project Challenges faced

While each project has its own unique characteristics, the context of this project had presented us both functional and technical challenges, the outcome of which is a satisfied client and creation of reusable knowledge at IT People.

- A major percentage of the previously developed test cases were not automation friendly and also needed modifications to reflect the new functional flow of information.
- Several types of users (like the Insured, Insurer, Beneficiary, Underwriter, Actuary, Broker and Assignor) exist and no test criteria for the access-controls were specified. This created an unidentified zone of security issues.
- The newly developed portal integrated sub-applications developed in different technologies to implement that Proposal, Underwriting, Policy Contract and Claims/Benefits modules. So, in certain test cases, more than one interface (developed in different technologies) was required to be exercised.
- The test parameters, and their possible values resulted in an exploded number of test combinations that needed to be tested to achieve a reasonable test coverage.
- The new web application has leveraged DHTML and AJAX, making it more challenging from Test Automation standpoint.

Our Solution

Test engineers at IT People spent considerable time to design the architecture of the solution before starting the development of individual automated tests.

Following were the objectives of the solution design:

1. Apply best practices in converting/upgrading the legacy tests to reflect the new claims-processing web application functionality and make the test descriptions automation friendly.
2. Create an easy-to-use test automation framework so that domain testers who are non-technical can also add new tests.

3. Leverage advanced techniques to use Mercury QTP without the Object Repository maintenance headache.

4. Effectively create load scenarios to identify the concurrency and response-time issues with LoadRunner, where each scenario simulates a business case.

5. Incorporate process-oriented test management, mainly on the test case management through Test Director.

6. Reduce the number of tests wherever possible while still assuring maximum test coverage.

The principal tools used in the project were all from HP Mercury, on which our engineers had sufficient exposure and our Testing CoE has supported the project through provision of various checklists, standards and test utilities.

The client expected minimal questions from the team pertaining to the Domain Expertise. We conducted an internal training to our team on general Insurance Claims-processing process from a Insurance domain expert from our Test CoE.

**High-level view of the solution:**

<table>
<thead>
<tr>
<th>Technology/Tools</th>
<th>Solution Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>HP Mercury TestDirector (for test management)</td>
<td>App Test Library</td>
</tr>
<tr>
<td>HP Mercury QTP (for functional test automation)</td>
<td>Utility library</td>
</tr>
<tr>
<td>VB Script (for external test tasks)</td>
<td>Test Driver Script</td>
</tr>
<tr>
<td>HP Mercury LoadRunner (for performance testing)</td>
<td>Subscripts</td>
</tr>
<tr>
<td>Stack of productivity tools like TestersDesk.com for Pairwise tests</td>
<td>Excel Data Tables</td>
</tr>
<tr>
<td>IT People Test CoE Assets (checklists, utilities etc)</td>
<td></td>
</tr>
<tr>
<td>Technology/Tools</td>
<td>Solution Architecture</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>HP Mercury TestDirector (for test management)</td>
<td>App Test Library</td>
</tr>
<tr>
<td>HP Mercury QTP (for functional test automation)</td>
<td>Utility library</td>
</tr>
<tr>
<td>VB Script (for external test tasks)</td>
<td>Test Driver Script</td>
</tr>
<tr>
<td>HP Mercury LoadRunner (for performance testing)</td>
<td>Subscripts</td>
</tr>
<tr>
<td>Stack of productivity tools like TestersDesk.com for Pairwise tests</td>
<td>Excel Data Tables</td>
</tr>
<tr>
<td>IT People Test CoE Assets (checklists, utilities etc)</td>
<td></td>
</tr>
</tbody>
</table>

On the Functional test automation aspect, our solution has included a tool-neutral test automation framework and leveraged the following features of Mercury QTP:
- Descriptive programming for functional areas that have a dynamic UI (therefore making it tough to follow the Object Repository based identification of the web controls).

- Automation Object Model (AOM) for automating the tool configuration and test initialization routines, in a way achieving automation-of-automation.

- External VB Script Library used as a project resource to perform functions like process monitoring, custom reporting etc.

- DataTable functionality to have the spreadsheet test data driven into the abstract tests.

The following two features of TestDirector were mainly used in the context of this project:

1. To document test requirements and map them to individual automation-friendly tests.

2. To perform test execution that involved multiple test machines within the test bed (some test cases had steps spanning two machines).

On the performance testing front, we first developed quantified test objectives (which previously did not exist) so that the tests are organized to validate the same. It was tricky to create realistic test data. Some background system functionality was dependent on the type of data and random unrealistic data was not an option in the generated load.

While the virtual user generators (VuGen) feature of LoadRunner was the significant element in preparing the transaction mix, other features like IP Spoofing (so that the claims-processing application treats all virtual requests as real requests). The execution semantics extended the traditional remote black-box style performance tests into analytical/intrusive style where several monitors were established to account for time spent at the client, network, app server and database server respectively.

The client needed a business view of the technical reports produced by LoadRunner. To achieve this, we had to export the test results into excel and create a ‘Response time Vs. Time Scatter’ chart and ‘Response time Degradation curve’ that are comprehensible to a typical business stakeholder.

**Results achieved**

**Costs saved:** The client, to our knowledge, has used the automation packs we developed for about 15 times on 15 builds within 6 months. Quantifying the costs of manual testing, the test automation’ ROI was justified, besides accelerating cycle time.

**Reusable assets:** Since we have engineered a framework supported and architecture savvy solution, a major part of the delivered artifacts were considered for reuse in other test automation projects of the client.
**Reduced tests:** By changing the strategy from random test selection to pairwise test coverage, we have achieved increased test coverage with decreased number of test configurations.

**Easier maintenance:** By virtue of making all the test data externalized into spreadsheets, the non-technical users, mostly business analysts, were able to study/test the system by easily changing the data that drives them.

**On-time deployment:** Because of the carefully engineered solution, and more importantly because of the relationship developed between the customer’s development team and IT People Testing team, all the operations got streamlined resulting in a timely deployment of the new claims-processing web application.

In conclusion, the solution was engineered to the best of our capabilities and has formulated a benchmark practice for the upcoming projects at IT people testing unit.