Test Automation is hard.

The open source CPAN TAF addresses typical complaints by establishing design principles.

<table>
<thead>
<tr>
<th>Challenge</th>
<th>Design Principle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lacks consistency between test cases</td>
<td>Reliable. Reusable. Repeatable. Universal.</td>
</tr>
<tr>
<td>Hard to debug</td>
<td>Independent and Adaptive.</td>
</tr>
<tr>
<td>Unpredictable</td>
<td>Web Interface for everyone.</td>
</tr>
<tr>
<td>Too complicated to run</td>
<td>Access from anywhere.</td>
</tr>
<tr>
<td>Hard to know execution status</td>
<td>Remote execution.</td>
</tr>
<tr>
<td>Difficult to expand</td>
<td>Real-time execution status and results status.</td>
</tr>
</tbody>
</table>

Other Impediments to Successful Test Automation

Treating It Just Like Development
Test automation's differences from development:
- Different life cycle.
- Different execution environment.
- Different methods of execution.

Misunderstandings
- Unrealistic expectations of ROI in early stages of automation.
- Test automation is viewed as a spare-time activity.
- Test automation is assumed to be simply record-and-playback.

Management Issues
- Test automation often runs over budget and schedule.
- Development team must buy in.

Staffing Issues
- Lack of QA-specific experience and skills.
- High turnover.

Gap Between Code Bases of Developers and QA
- Tests gets out of sync with code base.
- GUI testing tool is technologically behind GUI technology.
- GUI is unstable while being tested.

Test Suite Issues
- Tests are difficult to maintain and manage.
- Test results are hard to understand.

Flexible
- Supports a wide range of approaches to test suite creation.
- Supports management of different types of tasks.
- Open source allows for expansion by any developer.

Platform & Language Independent
- Cloud based: execution and access is distributed.
- Runs on: Windows OS, Linux OS.
- Automation instructions can be written in any language.
- Java, C, Python, Perl, shell, Expect, etc.

Extensible and Integratable
- Supports other automation tools.

Easy
- Easy to set up.
  - Easy to install from internet.
  - Peer-to-peer — server is optional.
- Easy to run and view results via web access.
- Easy to develop test suite.
  - Quick start development of test cases by following test bed examples.
  - Copy and paste hidden code from web page in UI.

Collaborative
- Supports different types of users by providing different interfaces:
  - User mode (web UI) for Developer and Tester and Manager.
  - Developer mode (command line) for QA Engineer creating automated tests.
- Share execution controls and test results anywhere in the world.

A Dashboard for Execution and Results Reporting
Web access (cross-browser) to a shared dashboard.

Execution Tools
- Start and stop execution of test suite or individual test cases.
- Pre-record day-to-day tasks and schedule automatic launch.
- Supports repetition of test runs.
- Supports longevity testing (24/7 execution).

Results & Status Reporting
- Captures test runs’ timestamp, duration, pass/fail, logs and related web resources.
- Displays real-time status.
- Saves and displays tests' historical information.

An IDE for Test Case Development
QA Engineers can develop complete test suites.
Start with the test bed — a batch/shell script that generates generic test suite or test case. Copy and paste code provided by IDE.
Use it as an educational tool: learn by doing and own the suite.

Future Development

Goals for our Open source Developers
- Remote launch via GMail.
- Integration with Google Earth.
- Integration with Amazon Web Services.
- Results delivery via email.
- Support for integration with more frameworks.
- Enhance user experience by using HTML5.

Designed for Real-World Workflow

FOR BUILDING AN OPEN SOURCE FRAMEWORK
by establishing design principles.

The open source CPAN TAF addresses typical complaints
Test Automation is hard.

Designed for Real-World Workflow

Try it! Simply Google “CPAN TAF”