Insurance Technology Services

# How to Improve Your Bottom Line with Proven Regression Testing Best Practices

September 2022

## Table of Contents

## 1 Executive Summary

Change is constant, inevitable, and necessary. Change also introduces risk. In the context of a system, one type of risk introduced by change is a system defect where there was none before, i.e., a regression. The objective of regression testing is to find these defects early and minimize their cost. An effective regression test plan provides the framework to allow for the most effective use of resources based on the available time, level of system change being introduced, and quality standards of the organization. Developing a repeatable process will not only improve the quality of your systems, it will reduce the costs and work effort of your ongoing production support.

## 2 Introduction

Insurance Technology Services has been helping the insurance industry successfully implement new systems for over 30 years. Our testing services group has seen it all when it comes to the variety of approaches companies take towards addressing the risk of introducing a disabling system defect into their production environment. Some choose to ignore the risk and hope nothing too bad happens. Others attack the problem with an abundance of effort and complexity that is counterproductive to the bottom line. In the sections that follow, we will reveal our answer to this Goldilocks problem with our proven regression testing best practices that are “just right.”

It is important to recognize that regression testing is different than user acceptance testing (UAT). UAT is focused on testing unique new features and fixes in a release while regression testing is more operational, involving discreet, repeatable processes.

## 3 Define what is important to test

The basis of an effective regression test plan is the scope definition. Before you can start developing a regression test suite you must define the scope of coverage. For an enterprise insurance system, the scope should cover the most common lifecycle scenarios across all products, states, and functional areas guided by the 80/20 principle. Twenty percent of a system’s functionality is likely to be used eighty percent of the time. Thus, highly used processes, functions, screens, transactions, menus, and fields should be the first candidates for coverage in the development of the regression test suite.

ITS has developed a proprietary Regression Test Checklist for insurance systems with a pre-defined and prioritized list of test conditions and scenarios that should be checked to determine if any significant regressions were introduced in a product, state, and/or functional area after delivery of a new system release. Our checklist template also includes traceability to the test cases within the regression test suite to enable accurate measurement of test coverage of a regression test run.

## 4 Develop a reusable regression test suite

Once you have a well-defined scope you can begin developing test cases that provide the necessary coverage to test what’s important.

Regression test cases should be designed in such a way that a minimum number of test cases provide the maximum coverage. However, this design approach should be balanced with the DRY (Don't Repeat Yourself) principle which advocates for smaller, discrete test cases that are easier to maintain and execute and are reusable across a variety of scenarios. Test cases should be organized into logically cohesive sub-suites rather than running a single, large regression test suite each time. This allows different subsets of tests to be executed when there is limited time or where there is confidence that only certain test cases need to be run.

After the initial regression test suite is developed it is important to keep it up to date based on any new enhancements and defect-prone areas. Review the notes of each new release to determine if your scope definition needs to be updated and enhance the regression test suite as needed.

Keep benchmarks on how long it takes to conduct your regression test runs. If they get too big and long, it can delay your release cycles. As new test cases are added you may have to consider removing old cases in areas that appear to be stable. Structure the regression test suite by coverage area (product, state, and functional area) so you can choose to run or not run certain areas as needed. However, you should always maintain a baseline that covers a wide scope across all the major coverage areas.

# 5 Automate

Regression test cases that are run frequently and take a long time to execute and verify manually should be automated. The effort to develop an automated functional test (AFT) is greater than what it takes to execute the same test manually. Therefore, many companies never get around to it. The return on investment is realized over time. However, without proper planning, tool selection, and maintenance you may never realize the payoff. The key to building an effective AFT solution is to first understand the common pitfalls to avoid.

Unexpected Complexity

Test scenarios must reflect the real-world production environment in order to identify potential problem areas. Business analysts typically know best what needs to be tested; however, these individuals are seldom equipped to address the technical challenges which can accompany the automation of complex test scenarios. On the other hand, technical analysts who understand the underpinnings of modern, web-based applications often don't understand the business well enough to develop a comprehensive test suite that provides adequate test coverage. Thus, the resulting AFT scripts are typically simplistic or error-prone unless both resource types are assigned.

Lack of ROI

Unfortunately, many AFT projects do not achieve the expected return on investment (ROI) due to the time and cost of the overall implementation, lack of test coverage, and, most significantly, maintenance issues. Without a reliable solution, the smallest system change can have a domino effect which breaks multiple AFT scripts and impacts downstream systems or processes in the insurer's production environment. All of these factors are influenced by the architecture of the AFT solution, which determines not only the level of reusability, but the reliability of the solution as well.

Tool Selection

When choosing an AFT platform, you should consider the following:

1. Can it support the complex test case scenarios and technical hurdles inherent in enterprise, web-based insurance systems?

2. What are the costs and availability of the required resource types and skill sets?
3. Does the architecture provide a high degree of reusability to drive down the development and maintenance effort?

ITS has developed the ITS Test Automation Platform (ITAP), a proprietary tool that redefines how AFT suites are built and maintained and dramatically improves the ROI equation. Highly efficient test scripts can be developed that intuitively segregate the intricacies of the test automation from the test information, enabling the reuse of common modules across a wide spectrum of test scenarios without the need for a high degree of technical acumen. The result is faster development of credible test scenarios with fewer resources required and lower ongoing maintenance costs.

## 6 General Best Practices

### 6.1 Grow your test automation strategy in stages

1. Develop an early win that provides real benefit, then add to it
2. Start broad by first developing a Smoke Test Suite
3. Then go deep in order of priority

### 6.2 Test what's important, not what's easy

Create a list of critical business showstoppers to drive your Regression Test Checklist. Examples may include:

- Policies issued with incorrect rates
- Errors on legal documents—policy forms, declarations, cancellation notice, etc.
- Rules that restrict users from completing business
- Submission workflow that slows or stops business issuance
- Missing underwriting restrictions
- Unexpected or incorrect ACH withdrawals
- Incorrect information on insured invoice
- Commission errors that are not corrected before reaching agents
- Inability to reserve claims at correct levels or against the correct coverages for statistical and financial reporting
- Claims workflow that impedes claims process or sends incorrect information to claimants
- Critical reporting data missing from stat tables

## 8 Definitions

1. **Regression**—a system defect introduced by a change that causes a failure in a function that previously worked before the system change
2. **Regression Test Plan**—defines the scope, approach, resources, and communication plan for the regression test process
3. **Regression Test Suite**—a collection of test cases (both manual and automated) designed to cover a defined set of test conditions and scenarios
4. **AFT Suite**—a collection of Automated Function Tests (AFTs) designed to cover a defined set of test conditions and scenarios. The AFT Suite is a subset of the Regression Test Suite.
5. **Smoke Test Suite**—a subset of the AFT Suite designed to quickly test a new release to determine if it is ready for manual testing

### 6.3 Know what is new

- What new features have been implemented in the new release?
- What bugs have been fixed in the new release?
- Concentrate your manual regression testing in these areas for these onetime tests. Before beginning, ask "What do I need to see to feel comfortable signing off?" such as:
  - ➢ System interaction with user
  - ➢ Changes in the database
  - ➢ Output

### 6.4 Real-world data & scenarios

- Use actual data and policies, claims, etc. from production
  - ➢ Take a (problematic) production policy and recreate all policy change transactions, claims, and billing cycles
- Test with realistic roles
  - ➢ Don't test as an admin user unless you are an admin user in production
  - ➢ Log in as an agent, underwriter, and other roles
- Test transactions as they are run in production
  - ➢ Most renewals, cancel notices, reinstatements, etc. are processed through batch jobs in the daily cycle - don't test these using any manual workarounds, run the daily cycles

## 7 Conclusion

Incorporating an effective regression testing process into any technology implementation provides a powerful safety net by mitigating errors introduced by new development. Efficient and effective testing dramatically reduces the possibility of project failure when a change affects a large portion of an insurer's production environment. By reducing the demand for testing resources and allowing broader test coverage, an effective AFT solution can improve the quality of the final product in production and save insurers from the pain of system implementation failure before it happens.