CastleBay Companies

Guidewire API Testing: GFIT – What It Is & What It Does

GFIT Introduction & History

The Gosu Framework for Integrated Tests (GFIT) was developed by Dr. Bill Shaffer, a long-tenured consultant with CastleBay. It is a Gosu implementation of the Framework for Integrated Tests (FIT) for Guidewire's InsuranceSuite (ClaimCenter, PolicyCenter and BillingCenter).

There is currently five Guidewire clients that are GFIT users. The first implementation, in 2010 was in support of a large ClaimCenter implementation. Recent implementations are for full InsuranceSuite implementations.

Advantages of GFIT

GFIT provides an extensive suite of fixtures that create data and execute tests with much higher performance than UI-based automation tools (e.g., Selenium or QTP). Additional advantages include:

- Easy test writing
- Low test maintenance
- Low test execution failure rate
- On-demand execution from personal workstations
- Fully automated execution from a Jenkins server
- Cost-effective customization for P&C insurers of any size.

With these compelling advantages, GFIT effectively replaces UI automation testing tools for server testing and data creation.

GFIT Performance vs. UI Automation

Table 1 summarizes GFIT's performance, which is 10-20X faster than UI automated testing. The claims contain incidents, exposures, and payments. The policies have multiple transactions.

ClaimCenter Claim	15 secs
PolicyCenter Policy	5 secs
BillingCenter Payment	8 secs

Table 1. Average time required to create a claim, policy, and account payment with GFIT.

Execution metrics are only a part of GFIT's advantages. Tests can be easily written by anyone with basic knowledge of an InsuranceSuite product. Since GFIT test data is captured in simple HTML webpages, any WYSIWYG HTML editor can be used. Test files are stable, and a new test

can be easily created by editing an existing test. In contrast, UI test scripts require tool-specific skills and must be continually updated as the UI changes.

At one CastleBay customer, a simple Java Swing client allowed rapid execution of tests in any environment, leading to widespread usage. Because the client also ran from the command line, it was deployed to a Jenkins server for fully automated execution. This simple configuration enabled test runs that rarely failed, and it produced an XML report in Junit format so Jenkins could display pass/fail results with automated notifications.

In comparison, access to UI automation tools is more restricted, test runs must be scheduled in advance, and are so slow that test frequency must be reduced. Expensive, complex configurations inevitably result in frequent test run failures.

GFIT vs. UI Automation Architecture

Figure 1 compares GFIT vs. UI-automation architectures. Both techniques access the server's public API, but tools like Selenium are limited by UI response times, resulting in slow performance. The GFIT library is a facade API to the server API. This facade is organized around fixtures which encapsulate tasks, and each fixture accesses the server API to complete its task. GFIT clients access fixtures via a SOAP web service, and execute the tasks listed in the test document using the data provided.

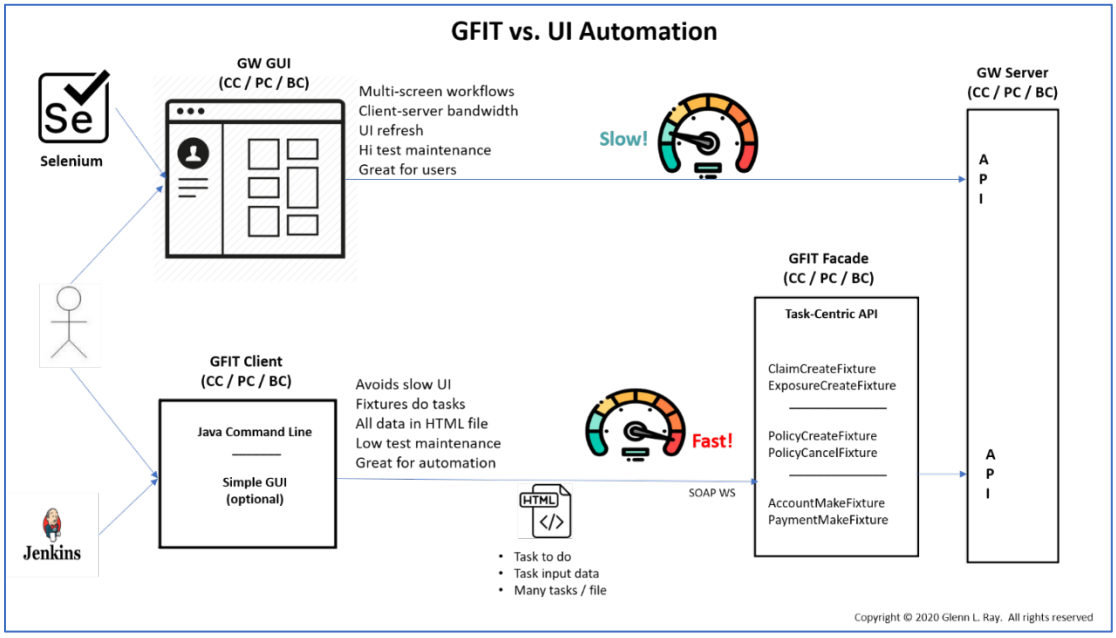

Figure 1: GFIT vs. UI Automation Architecture

CastleBay Companies

Summary and Conclusion

GFIT is a critical component in an efficient DevOps CI/CD pipeline. With the ability to rapidly create data and execute tests, it either replaces time-consuming, manual QA efforts, or outperforms UI automated testing by 10-20X. The resources freed up can be allocated to more productive tasks, resulting in higher project velocity and lower defect rates. GFIT is a breakthrough QA automation solution for Guidewire clients.