Choosing a License for Your OpenTAP Plugin

The Open Source Path to Effortless Automation

This white paper examines the licensing and distribution options available to developers, publishers and distributors of plugins for the OpenTAP test automation platform. These plugins are created by the gamut of OpenTAP ecosystem participants – OpenTAP users, integrators and other third-parties, as well as by Keysight – and serve to expand the functionality of the platform and accommodate different hardware configurations. If you've created an OpenTAP plugin, it merits thoughtful consideration of licensing and distribution options.

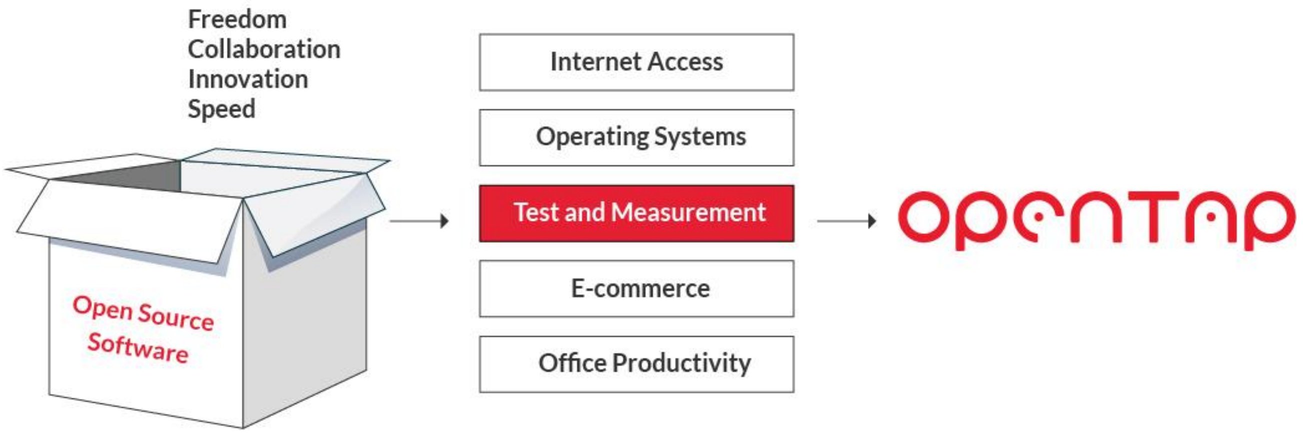


Table of Contents

Introduction

What is an OpenTAP plugin?

The OpenTAP platform architecture is very straightforward: the OpenTAP core engine handles scheduling and enables execution of tests, while OpenTAP plugins provide interfaces to devices under test and test instruments, encapsulate test steps, implement user interfaces and result listeners, and can even be used to support new languages (e.g., Python).

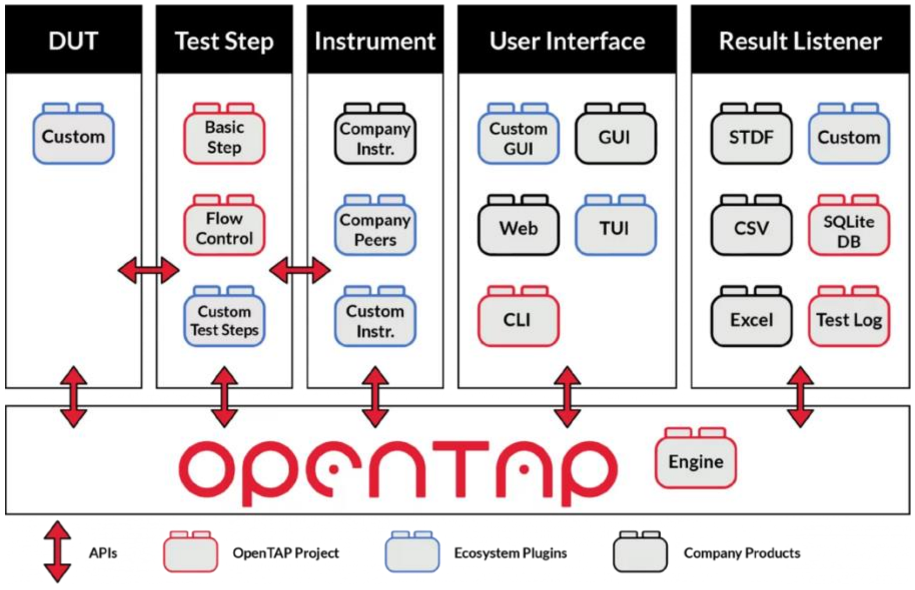


Figure 1. – The OpenTAP architecture and types of plugins.

The plugin categories highlighted in Figure 1. serve a variety of purposes:

DUT
DUT (Device Under Test) plugins cover a broad and varied set of use cases, with each DUT potentially entailing unique tests and test harnesses for different device types, configurations, connectivity, etc.

Test Step
Test step plugins and flow control enable sharing of new plugins from ecosystem participants and within development groups.

Instrument
Instrument plugins help instrumentation vendors foster greater interoperability for their new and existing equipment.

User Interface
User interface plugins accompany the core OpenTAP project. Keysight offers a professional-quality GUI for creating test plans, and other UI plugins are underway from ecosystem participants.

Result Listener
Result listener plugins convert test results into familiar document formats (CSV, Excel, etc.) input into SQL and other types of databases, or raw data.

Who Creates OpenTAP Plugins?

The OpenTAP project itself includes a number of plugins, but even more plugins come from OpenTAP ecosystem participants who create them to expand functionality and add value.

Users of OpenTAP also create plugins to contain the test steps needed to test their hardware and software.

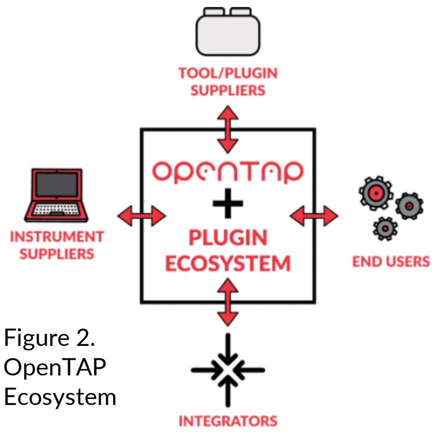


Figure 2. OpenTAP Ecosystem

Software Licensing Overview

Software, broadly speaking, is intellectual property realized as program code. Software may enter an organization from commercial vendors, contractors and/or from open source project distributions, or an organizations itself may create software internally. An organization may choose to retain software entirely for internal use; it may be used to offer a service to customers or partners; and it may be (re)distributed by the organization as a free-standing software product, as part of a software product or tool kit, or embedded in a device or other piece of equipment.

To govern acquisition, use and (re)distribution, most software today is released under one or more software licenses. Even software with no apparent license is still subject to copyright. Proprietary software, e.g., desktop utility software like Microsoft Word, is typically licensed under a "EULA" (End User License Agreement) that restricts usage to a single user or company site and prohibits redistribution and reverse engineering. By contrast, open source software licenses offer few restrictions and encourage redistribution/sharing and access to source code. But open source software is more like free speech than free beer: acquiring open source software may be free of monetary cost, but the particulars of usage and especially of derivation and redistribution are stipulated by the governing license.

There are over 60 open source licenses under the purview of the Open Source Initiative (OSI), plus several thousand more self-styled "free and open source" licenses that may or may not conform to the principles laid out in the Open Source Definition.

OpenTAP and Plugin Licenses

OpenTAP has been designed and licensed to provide the greatest possible flexibility to the developers of plugins. The OpenTAP core is released under the Mozilla Public License version 2.0 license which:

- Imposes no significant impediments or obligations to test and measurement users
- Encourages (but does not require) contribution of modifications back to the OpenTAP community
- Is compatible with other permissive open source licenses such as MIT and BSD

Licensing Considerations

Choosing a license for your OpenTAP plugin should not be a daunting task. However, there are key considerations you need to address regarding your plugin itself and your business and technical goals in creating and distributing it.

These considerations include

- Business goals
- Technical goals
- Dependencies on other software
- Status of the intellectual property involved

Some of these considerations are actually interlinked, requiring extra attention, and most cascade into sub-considerations. Let's examine them one by one.

Business Goals

Let's start by determining how your plugin will support your business.

Internal use only

End-users of OpenTAP typically develop plugins to contain test steps, describe devices under test (DUTs) and support legacy test equipment. If you don't plan to share your plugin with third-parties, you have maximum freedom in choosing a license and may elect only to make copyright claims for it. By keeping the plugin as internal-only, however, you will lose the opportunity for community collaboration around support and enhancement. The entire technical burden will lie with you and your organization.

Accompany/support another product, e.g., an instrument, device or another plugin

Instrument suppliers and device manufacturers who want to encourage use and ease integration of their hardware will want to supply OpenTAP plugins to their customers and partners, usually shipped together with their products or on their company web sites. For these use cases, plugin publishers have the greatest flexibility in choosing a license. However, if the immediate recipient of your plugin is also likely to redistribute it "downstream", you will either need to craft a proprietary license allowing such redistribution or choose a compatible open source license.

Sell as shrink-wrap software

You may create a high-value standalone plugin, e.g., a rich *result listener* or an *instrument interface* for legacy test equipment that you would like to monetize. The most obvious path is to distribute such software under a perpetual EULA, but there is nothing preventing you from creating and marketing commercial plugin software under a renewable subscription or an open source license.

Technical Goals

The most common technical goals are aligned with the business goals above – enabling product test automation and interfacing to test equipment and/or other plugins. Following are some additional, primarily technical goals for creating and distributing a plugin.

Addressing new test instruments

Both end-users and instrument manufacturers may wish to integrate support for new test equipment that is not currently supported by OpenTAP and existing plugins.

Extending functionality of an existing plugin

You and your organization may need to enhance or re-architect an existing plugin to meet your technical needs. Another scenario might be rehosting a plugin to a different execution platform, e.g., from Windows to MacOS or Linux. If the existing plugin is open source, you will be best served by enhancing that plugin through project contributions, or you can fork it wholesale and create a new plugin. But not all plugins are open source. Alternately, the plugin of interest may be open source but published under a license not approved by your legal department. In either of these cases, your best strategy may be to create a new plugin.

Adding completely new capabilities to OpenTAP

The OpenTAP architecture is highly flexible, with plugins the main path to extending functionality (vs. modifying the core engine). Possible new capabilities implemented as plugins include adding new language support (e.g., for GO or Rust), creating integrations with other software frameworks (e.g., Jenkins, or SCM and SCA systems or inhouse enterprise software) – the sky's the limit!

Fixing bugs in existing plugins

If you find bugs in an existing plugin, the easiest path is to report the bug to the plugin author or maintainer – this method applies to both proprietary and open source plugins. For open source plugins, it may be faster, to fork the plugin and create your own fix. Note though, that even if you also create a pull request, your patch may not be accepted, and you can find yourself maintaining that fork for the remainder of its life.

With both private and community forks, your organization must be comfortable with the existing plugin license, especially if you need to redistribute it.

Dependencies on Other Software

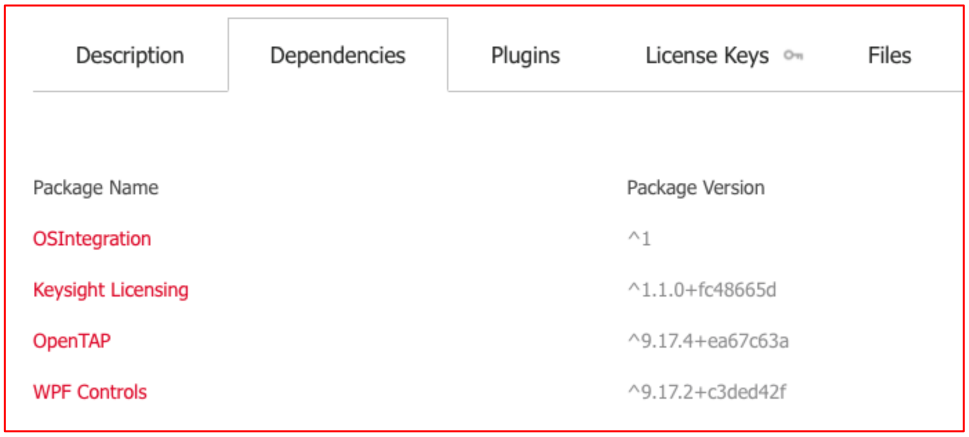
No man is an island, and no software exists in a technical vacuum.

All types of software, even when presumably written from scratch, have dependencies on other software components. OpenTAP plugins can have dependencies upon

- Existing plugins or templates used as starting points for your new plugin
- OpenTAP core engine APIs and .NET APIs
- Language run-time libraries (for OpenTAP, usually C# libraries)
- Existing plugins that your plugin may call to implement needed functionality
- Libraries that you may create to support multiple plugins or other software projects
- Other third-party libraries from SDKs, open source projects, etc.
- Nested dependencies, from the above-listed code upon yet other software components

To enumerate the dependencies for a plugin, you will need to review code and API usage in the plugin. Such a review can occur manually or preferably using Software Composition Analysis (SCA) tools that cross check against knowledge bases of open source project code. Some of those tools will also check for "snippets" – code fragments copied from open source projects or articles and pasted inline into your plugin.

Dependencies among packages hosted on OpenTAP.io appear as part of package metadata:



Package Name	Package Version
OSIntegration	^1
Keysight Licensing	^1.1.0+fc48665d
OpenTAP	^9.17.4+ea67c63a
WPF Controls	^9.17.2+c3ded42f

Figure 3 – Dependency list of the Timing Analyzer package.

If there are dependencies upon open source code, you will need to examine the licenses for those dependencies. Broadly speaking, "liberal" licenses – Apache, BSD, MIT, Mozilla and similar licenses – will require minimal compliance effort when distributing derived works. "Reciprocal" licenses, especially those in the GNU Public License family (GPL et al.) may require that you disclose part or all the source code to your plugin and/or license it under the same or a similar restrictive license.

Status of Intellectual Property

Does your plugin contain closely-held / proprietary intellectual property (IP)?

If your plugin contains closely-help IP – trade secrets, software patents and code implementing methods patents – you will need to work in concert with your legal department in advance of releasing. Issues include

- You may want to protect that IP by releasing the plugin as proprietary software or as the proprietary portion of an "open core" deliverable, wherein the commodity functionality is open source while the value-added portion is not,
- If your plugin contains patented material, you should consider making a patent grant or license for use of that IP in the context of an OpenTAP plugin license. Alternately, you should warn potential users of such patented technical material that your organization will require negotiation of a patent license for its use.

Do you intend to keep the source code proprietary within your company?

Even if the underlying OpenTAP platform is open source, there is no requirement for plugins to be released to any third parties.

Hosting and Distribution Scenarios

The OpenTAP developer community and the larger OpenTAP ecosystem (developers, users, partners et al.) benefit from the availability of plugins to support the widest possible range of test automation applications and scenarios. To encourage development and distribution of plugins, the OpenTAP project enables creation and distribution of plugins licensed as both open source software and/or as proprietary/closed source programs, hosted together with OpenTAP in the project repository or in other locations.

Plug-in License	**Hosted on OpenTAP.io**	**Hosted Elsewhere**
Open Source w/Approved License*	√	√
Open Source w/Other Licenses†		√

* Mozilla Public License, MIT and BSD
† Apache Public License v2, Eclipse Public License, LGPL, etc.

Proprietary/ Closed Source	Under Discussion	√

Figure 4. Licenses and hosting options

Following are concrete, real-world examples of how you are likely to create and distribute your plugin and your licensing options for each.

Creating and Contributing Community Plugins as Open Source

The OpenTAP test automation platform is open source software, so it makes sense that extensions to that platform also be open source (although it's not required). If you publish your plugin under a license approved for OpenTAP (MPLv2, BSD or MIT), you have the option of hosting it directly on the OpenTAP project site and repository, together with the OpenTAP core engine and with other approved open source plugins. Of course, you can always host or distribute such a plugin on your company site or repository, or together with your product, as well.

Creating an Open Source Plugin and Hosting it Yourself

The universe of open source licenses is large, and the choice of licenses is varied. If you ***choose*** a license not currently approved for OpenTAP, or are ***required*** to use one because of dependencies on reciprocally licensed software in your plugin, your hosting options are still very broad. You can host or distribute your plugin via

- your company web site
- your company repository on GitHub, GitLab or equivalent project hosting platforms
- direct distribution with your product or on removable media (CD-ROM, etc.)

It is important to understand that if your plugin license is incompatible with the OpenTAP platform license or with other software in a test automation configuration, you are effectively passing responsibility for resolving that incompatibility "downstream" to your customers or other recipients of your plugin.

Creating and Distributing Closed Source Plugins

This scenario, while conceptually appealing to many companies (especially their legal departments), is today an exceptional case. Given the scope and extent of dependencies on other software present in modern software), it is difficult to imagine developing and distributing a 100% proprietary plugin for OpenTAP. That said, if your organization has developed a proprietary plugin, and you want to share or distribute it, you have several options for making it available to the OpenTAP ecosystem:

- Host on your company's site or repository
- Distribute the plugin via a third-party site or marketplace

Hosting proprietary plugins is currently under discussion for the OpenTAP project site.

Enriching the OpenTAP Ecosystem

The OpenTAP ecosystem spans the gamut of end-users to integrators to test equipment manufacturers and beyond. New and incremental functionality from plugins is a key factor for making the OpenTAP platform attractive and helping test automation address the highly dynamic technology marketplace.

Take the first step.

Visit OpenTAP.io and join the OpenTAP Community.

- Learn about how to write and package plugins – Video: Creating OpenTAP Packages
- Check out a template for building plugins
- Explore the OpenTAP Project on GitHub