Embedded COMPUTING DESIGN

Eight ways to avoid bad end-user software licensing

KEN WHITAKER, VP, ENGINEERING, DATALIGHT

0 Comments - **Leave a Comment**

As the traditional software application market migrates from the enterprise to mobile and embedded devices, end-user license management has become a must-have design issue.

The emergence of the Internet of Things (IoT) has energized the embedded market and to some, it may appear to be an untethered maze of devices, each doing its own thing. The companies that provide these devices and publish the software running on them are hunting for ways to keep track of authorized use, whether it's industrial, mobile, or medical devices, or even wearables. There needs to be a way to employ technology that limits unauthorized access of the embedded device, apps, and especially third-party apps. License management with embedded systems can be quite complex, far beyond traditional user names and passwords logins. And this takes some pretty intelligent software.

Licensing software components are either home-grown or available as purchasable software components. Common services that end-user licensing needs to manage software apps include:

Requesting personalization, such as user name/password or machine ID

Verifying that the installation is authorized (and if not, refusing to run the app) for use, update, and support

Allowing full use, a temporary trial, or specific features to be enabled

Optionally, providing an interface for audit of app use, statistics, and other interesting patterns identified by the end-user (or device)

There are many software tool providers who license management components. For example, SoftwareKey System provides this capability on a variety of platforms, including desktop operating systems and even Linux running on many of today's embedded devices.

How to define end-user licensing

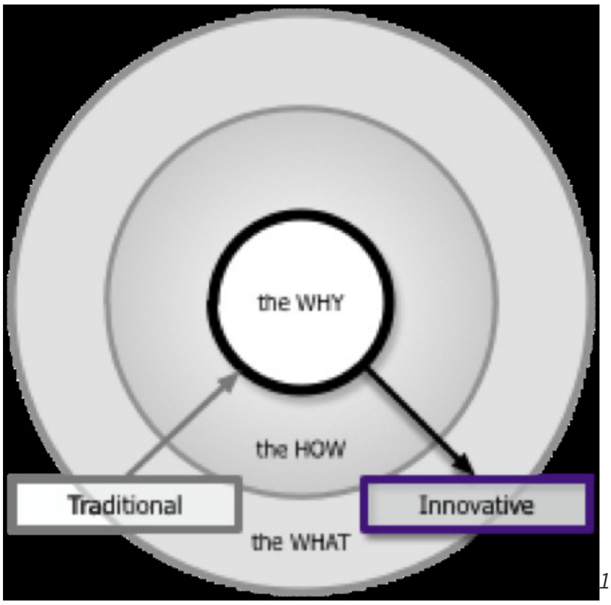
It's easy to superficially explain the reasons for end-user license management. Most anyone in a software development organization can explain *what* they do for end-user license management. *How* software is

NEWS

Linux

- Feel "free" to use embedded Linux — 4 weeks ago
- New "Accel64 for Linux" Software Supports 64-Bit Systems — 2 months ago
- Excel Software Adds QuickLicense Support for 64-Bit Linux and Xojo Applications — 5 months ago
- prpl Foundation Launches First-Ever OpenWrt Summit 2015 in Dublin — 8 months ago

licensed is also straightforward for some team members to explain. Understanding *why* is far more important. One expert explains the importance of defining a product, service, or software licensing as *the why* before *the what* and *the how*.



1. Innovative companies start with the why *and then define the* how *and then the* what, *not the other way around.*

The actual terms of the license are summarized in something called an *end-user license agreement* (EULA, for short) that binds you to specific rules you must agree to before using the software. Oftentimes, EULAs tend to be so wordy and confusing that most people don't even bother digesting the pages of "thou shalt not" legalese. The EULA represents the *what*.

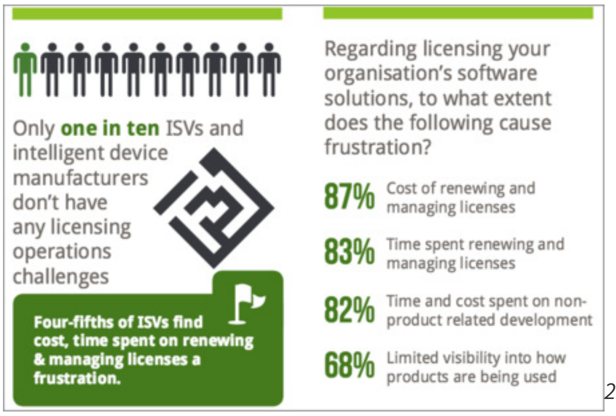
Fewer people in a software development organization can explain *how* end-user license protection works and is enforced. End-users quickly find out the licensing issues when they're locked out from using the software. The software techniques used to enforce the license represents the *how*.

For end-user license technology, the *why* defines how aggressive the software provider wants to avoid being ripped off from bad users. The techniques can be extremely rigid to ensure that the company is paid for every copy being used. That is more of a *result*.

I propose that the w*hy* behind software license management should provide the guidance that results in a win-win between the end-user and the software app provider. Here's an example of a well-articulated *why*: "We trust our customer and authentication is required to keep the software under maintenance and to provide the best technical support possible. Unless unauthorized users are knowingly misusing the software, it's the company's philosophy to not restrict use until the software is upgraded."

Some interesting stats

It appears that software vendors, overall, are failing their customers with unreasonable software licensing methodology. In fact, the time spent dealing with licensing for paying customers should be close to nothing. Apparently I'm not alone in this view. Figure 2 highlights statistics from a recent technical community survey hosted by the Gemalto Group.



Only **one in ten** ISVs and intelligent device manufacturers don't have any licensing operations challenges

Four-fifths of ISVs find cost, time spent on renewing & managing licenses a frustration.

Regarding licensing your organisation's software solutions, to what extent does the following cause frustration?

87% Cost of renewing and managing licenses

83% Time spent renewing and managing licenses

82% Time and cost spent on non-product related development

68% Limited visibility into how products are being used

2. Software licensing frustration abounds both with the software vendor and the customer.

According to the Gemalto survey, they found that only one in ten independent software vendors (ISVs) didn't have tough software licensing logistics to deal with. This means the other 90% struggled with some aspect of licensing. For customers who were involved with software licensing, more than 80% were frustrated with the cost and time spent managing that licensing. Gartner estimates that more than $400 billion was spent on software licensing worldwide last year, with between 5% to 6% of the total amount generated by audits.

I recently had the "pleasure" of attempting to resolve a simple multi-user software licensing issue only to require intervention from several customer support specialists. In this case, the problem wasn't properly resolved and we ended up switching software vendors. The software supplier lost a customer as a result.

Rules of the road

Like so many technology providers claiming that the obvious mantra of "the customer is #1," nothing could be farther from the truth. There are common-sense approaches that we, as software developers, can put into place to make legitimate software use effortless to deploy, simple to authenticate, and easy to use. Here are my top eight pet peeves:

1. Assume that the customer is a bona fide user. In other words, *trust* the customer. If a customer is having a hard day and a password is repeatedly entered incorrectly, don't lock the user out.

2. Simplify the EULA. You shouldn't need a lawyer to understand your rights to use the software. If you don't know how to simplify your EULA, use the Hemingway app to help simplify your text. Best of all, their online version is currently free to use (and EULA-free).

3. Trial periods are for losers. Restrictive software use under a trial period is simply bad business. It's even worse if you lose the ability to access data you created after an expired trial period. Many vendors make it impossible for a well-intentioned customer to take enough time to try the software out. Software vendors need to ask themselves, are you trying to earn the right to add a new customer or are you putting unnecessary time pressure to force a decision?

Datalight's evaluation software is fully functional, and though the initial trial period is 30 days, a simple phone call to your account manager can extend it if needed. And all of the work you do in the evaluation passes seamlessly to the fully licensed version.

4. Abrupt termination of license. I've personally experienced a situation when a valid licensed product somehow becomes illegal, seemingly out of the blue. Even though I owned a legitimate copy, I was challenged by a support specialist insinuating that I had not made a legitimate purchase. I had valid license proof, but I felt on the defensive—not like a valued customer.

5. Exceeding maximum installations. This is a big one! Software developers are getting smart with development of apps that can be installed on more than one device with a single user login (usually an email address). The modern computer user has a couple of desktop computers, a laptop, and a smart phone. As long as the end-user isn't using more than one device at a time, why do you even care? Let customers install the software on as many devices as they own.

6. Oh no, there's no Internet connection. Some corporations don't allow systems behind a firewall to have wide-open Internet access. This is even true with industrial embedded devices in a factory where wireless access is spotty at best or with mobile devices where the device loses wireless coverage. The software license management code that periodically does a background check for a valid license over the Internet will fail and, in some cases, the app stops working.

7. User identification. Make it possible for the user to identify some unique identifier that can be used for support with the vendor in case of trouble. How often have you clicked on an app's "about" window to view your serial number only to be told that it's a non-existent, obfuscated identifier that can't be used by the vendor's customer support.

8. Make it possible to transfer licenses. Let's face it. As employees are hired (or leave), tying a software license to specific names can be asking for trouble. Being able to transfer a license to another name is an absolute necessity. We've all found creative ways to get around it by either using a bogus or generic email address like support@companyZ.com. That is just asking for confusion down the road.

Solutions to fair licensing

The Trusted Computing Group (TCG) is a non-profit organization comprised of technology member companies with the intent to promote standards for supporting end-to-end trusted computing environments on any platform, including embedded systems and IoT devices.

Innovations are already being made with the advent of software and software with special ICs to ensure that trusted software is licensed only on specific system. Wibu-Systems, for example, has announced CodeMeter License Central, which provides a secure licensing platform via special hardware that can be deployed on desktop systems, mobile devices, factory PLCs, memory cards, and embedded systems. This hardware/software solution places licensing logic *behind the scenes* so that the end-user (or for those devices without a user interface) doesn't have to be involved with authentication. Once the device is properly authenticated, the software just works. The encoded hardware does the work and the user doesn't have to.

Yet another organization, Campaign for Clear Licensing, has been created to solve this problem. Mark Flynn, their CEO, says it best. "We believe end-user organizations need effective representation because everybody wants to be legal, everybody wants to invest in software. But often licensing agreements are incredibly complicated. You often need a law degree as an IT person or a buyer really to understand them."

Should history repeat itself?

In the 1980s, commercial desktop software provided all sorts of end-user safety checks that were close to vengeful. I remember one software company that would start a countdown timer. After ten seconds, it would automatically reformat your hard drive for attempting to use an unauthenticated copy of the app. To add insult to injury, as the software was counting down, it displayed a bomb falling on the video monitor until it explodes.

As I mentioned, EULAs have become too complex for the average user to comprehend. Years back, Borland International came out with a one-page no-nonsense EULA that was a work of art. It was simple, written in a way that was easy to understand, and actually gave the reasons why a license was needed in the first place. Why haven't more software development vendors learned from Borland? Software developers servicing the embedded world need to trust the customer and employ license policies that benefit everyone.

Ken Whitaker is the Vice President of Engineering at Datalight, where he leads the engineering, QA, technical support, and project management functions. He has degrees in Mathematics and Fine Arts from James Madison University and advanced Computer Science coursework towards a Masters degree from Virginia Polytechnic University (Virginia Tech). Ken is a frequent presenter at industry events and has authored two books: Managing Software Maniacs and Principles of Software Development Leadership.

THIS ARTICLE WAS PUBLISHED ON MAY 5th, 2016.