# Reassessing the Use of Commercially Licensed Qt

**By Mark Hatch, COO**
**Integrated Computer Solutions, Inc.**
**March 28, 2011**



**54 B Middlesex Turnpike • Bedford, MA • 01730 • www.ics.com • 617.621.0060**

## Reassessing the Use of Commercially Licensed Qt



*Copyright © 2011 Integrated Computer Solutions, Inc. All rights reserved. This document may not be reproduced without written consent from Integrated Computer Solutions, Inc. Qt, Qt Software and their respective logos are trademarks of Nokia Corporation. All other trademarks are property of their owners.*

# Reassessing the Use of Commercially Licensed Qt

## Introduction

Today, over 400,000 developers use the Lesser GNU Public License (LGPL) version of the Qt® C++ Framework. Furthermore, some 3,500 companies have purchased a license of Qt under Nokia (or Trolltech's) Commercial License Agreement. Although the number of companies that are strictly using the commercial version of Qt has dropped dramatically over the last several years, there continues to be a significant number of companies that are willing to purchase new commercial licenses and support every year.

The motivation of these companies to pay for the exact same bits of software that others use for "Free" may be perplexing to some proponents of open source. However, there have been some very real business issues driving these companies to purchase the software, including needs to:

- Modify license agreement
- Establish direct relationship with Nokia
- Deploy statically linked applications
- Satisfy regulatory requirements
- Avoid the risk of using the LGPL

The ongoing validity of these business reasons to purchase commercial Qt licenses may have changed on March 7, 2011, when Nokia announced that it was divesting itself of the commercial Qt licensing and support business. This divesture included the reassignment to the successor company of a total of 19 people, primarily the Qt worldwide sales and marketing staff. It did not include the existing Nokia front line support engineers, nor did it include any of the 200+ engineers who are working on extending Qt and fixing bugs. During the transition, Nokia will continue providing these services but in the longer term, the acquiring firm is expected to hire and train their own engineers to do this very technical work.

This divestiture directly benefits customers because it opens the door for other suppliers to offer Qt support and compete based on price, expertise, geography, and service levels. Since all support suppliers will submit their changes to Nokia via the new Open Governance process, it is a level playing field where companies will compete for business.

The purpose of this paper is to re-examine each of these business issues and see whether the divestiture might require a reassessment of a company's decision to use the commercial version of Qt instead of the LGPL version.

## License Agreement Modifications

Open Source license agreements, such as LGPL, are by nature nonnegotiable. You either use the software under that license or you do not use the software. However, a company’s Legal or Risk Management Department often defines a minimum set of license rights or

*Copyright © 2011 Integrated Computer Solutions, Inc. • www.ics.com • Phone: 617-621-0060*

warranties that it finds acceptable. As one could imagine, a Legal department or Risk Management organization could identify many different issues. However, in talking to customers, the most common requirement is that any software used in a mission critical application must include "indemnification against infringement of another company's intellectual property."

Ignoring the philosophical arguments about whether software is really intellectual property, pragmatically there are plenty of multi-million (or even billion) dollar lawsuits every year claiming that one company is illegally using the intellectual property of another. Sometimes these lawsuits are based on very objective criteria (i.e., does Company A's source code include any subset of Company B's source code). In other cases, the decision is much more complex, dealing with whether the software leverages another company's idea or concept (often patented). So when a company requests indemnification against infringement, they are requesting that the supplier stand behind their code and make things right (typically by replacing the offending code or buying a license from the suing company). Software is expensive, and a company does not want to be prevented from using that investment while two other companies wage long legal battles.

The value of this clause really depends on the ability of the company that is doing the "indemnifying" to deliver on its promises. In the case of Nokia, with its annual revenue of greater than 40 billion Euros and some 200+ engineers that focus exclusively on the Qt framework, most people would agree that this is a strong warranty. Basically, any infringement action that Nokia cannot overcome legally, its engineers can re-engineer out the offending code. In the worst case, Nokia could just buy a license to use the infringed company's intellectual property of its customers.

The strength of this promise of indemnification changed significantly when Nokia decided to exit the direct sales of commercial licenses and support and assigned its existing contracts to a third party with less than 0.2% (that is two tenths of a percent) of Nokia's annual revenue and none of the core Qt engineering staff.

Interestingly, the open source version of Qt might be more resilient in this new environment for two reasons:

1. There is no central place documenting companies that use the LGPL version of Qt. This means that an infringed company will have to do its own research to identify Qt users, or offer an attractive price for a license to use its intellectual property so that companies will voluntarily identify themselves to eliminate any potential liability.
2. Open source tends to evolve much more quickly than commercial versions with well-defined (and lengthy) development cycles. It is certainly possible that open source developers could re-engineer the offending code, test it, and distribute the patches worldwide before the commercial version has even set a release date.

*Copyright © 2011 Integrated Computer Solutions, Inc. • www.ics.com • Phone: 617-621-0060*

Given the changed situation, companies that have purchased commercial licenses or support in the past based on the availability of indemnification might want to reconsider the risk/value equation to see whether it continues to make business sense.

## Direct Relationship with Nokia

Most people have that special restaurant or bar where they are a known customer and thus get extra special treatment and concern by the proprietor. Similarly, some executives of companies that are investing hundreds of thousands or millions of dollars of engineering based on Qt want a special relationship with Nokia that allowed them to have special treatment if they ever needed a bug fixed or a specific enhancement. Obviously, there is no guarantee of action, just a guarantee that they would be heard by someone. This is a substantially different outcome than a company that was using the LGPL could expect if they ran into a critical problem and had no prior relationship with Nokia.

This also changed with Nokia's divestiture of the sales and marketing of Qt. The direct relationship was transferred to a third party and there is no direct relationship possible with Nokia.

Companies that have been purchasing commercial licenses and/or support for Qt on the basis of having a direct relationship with the original technology provider, might want to consider whether the indirect relationship that now exists offers the same (or even any) value.

## Deploy Statically Linked Applications

Although the Free Software Foundation would prefer that everybody used the basic GNU Public License with its commercially difficult clause requiring the sharing of potentially proprietary source code, they pragmatically recognized that this was not possible for all software. Consequently, they created the LGPL license for libraries and many companies developing proprietary products have found it acceptable over the years.

However, the LGPL still incorporates a key "freedom" for end users – the ability to upgrade or replace a library provided under the LGPL even in deployed applications. (An important footnote is that the LGPL 2.1 license used by Qt does not require that the application be actually functional if the library is switched. This loophole is often referred to as the "TiVo loophole" and was eliminated in the V3 of the LGPL.) The most obvious benefit of this freedom is that a new library might have improved performance or a key security patch. However, the library might have also been purposefully hacked to expose proprietary communications and allow their exploitation.

In most modern operating systems, libraries are dynamically linked into an application at runtime providing a number of technical benefits. But in regards to the LGPL license, this dynamic linking also satisfies the requirement that users be able to replace the Qt library with a different version.

*Copyright © 2011 Integrated Computer Solutions, Inc. • www.ics.com • Phone: 617-621-0060*

There are several situations where applications cannot be dynamically linked. Most notably in the case of real time applications or applications where there are regulatory requirements that forbid "experimentation" (e.g., medical applications). In these environments, application developers must build "statically linked" executables where one executable file includes everything needed to run the application.

Statically linked applications are more challenging to create while maintaining compatibility with the LGPL because a company needs to provide users with the ability to re-link. In the classic "hello world" example, this is trivial to satisfy – basically it is a single ".o" (compiled, but not linked) file along with a single command line for the linker. However, for almost any non-trivial application, enabling a user to successfully re-link the application is a very difficult task often involving specialized scripts, multiple archive files, and quite frankly a little bit of magic. The complexity of abiding by the LGPL with a statically linked application has resulted in Nokia asserting that statically linked applications could not use the LGPL and that the commercial license agreement was the only option. Although this is incorrect at a theoretical level, pragmatically, Nokia was not far off the mark.

It is possible that companies that must statically link their application have no choice but to purchase commercial licenses for their developers. However, before assuming this, it is best to ask the developers. Remember the application does not have to run (and perhaps that should be a built in goal in some cases to prevent meddling), it just has to link under the LGPL V2.1 license. You also do not have to describe how to install the new executable on a device. So if the device is locked down (e.g. burned into ROM), then there is little danger of mischief.

## Satisfy Regulatory Requirements

Some industries, most notably in the health and safety areas, require careful tracking of a source code's provenance to ensure that bugs or exploits have not been (in)advertently injected. Not surprisingly, in industries with these types of requirements, commercially licensed software is assumed to be better controlled and tested than open source code.

This belief on the surface seems reasonable, and might have been correct in the Trolltech days when the commercial version of Qt code was the base for development that was later made available under the GPL. However, today the base assumption that commercial comes first and then open source is released is fundamentally flawed. The current release of Qt, 4.8, that is being developed at the point when this paper was written, is a public source code repository with contributors from around the world. It is perfectly correct to observe that the LGPL version of Qt comes *first* and the commercial version is later derived from it.

Therefore, companies that are required by regulatory requirements to trace the changes in the source between versions derive no significant benefit from using the commercial version of Qt because is based on the exact same source code as the LGPL version.

*Copyright © 2011 Integrated Computer Solutions, Inc. • www.ics.com • Phone: 617-621-0060*

Companies that have been purchasing the commercial version of Qt because they believed that they have greater traceability of changes should review their decision and confirm that this still makes business sense.

## Avoiding the Risk of Using the LGPL

In some companies, the use of open source software is strongly discouraged or even forbidden because of fears that an accident could result in the loss of the company's intellectual property. And since Qt is available under both commercial and open source licenses, some companies simply choose to purchase the commercial license to reduce that risk.

The idea that this is a "safe" choice is incorrect. First, "commercial Qt" includes a number of subsystems, including WebKit, QtScript, Phonon, PySide and Qt Help that are actually provided under the LGPL. And the number of these subsystems that will be provided under the LGPL will definitely increase over time as Qt is extended to embrace more APIs. Consequently, a company that is purchasing and using the commercially licensed version of Qt might already have the full liability to abide by the LGPL license. Although it could be argued that this is a matter of degree, the fact is that if you use Qt, you need to be aware of the obligations and put the proper processes in place to avoid any errors that could result in the loss of intellectual property.

Secondly, even if you don't use the LGPL portions of commercially licensed Qt, your application almost certainly use other libraries that are only available under the LGPL. For example, the C Standard library on Linux is provided under the LGPL. Other popular libraries include FFmpeg, MySQL Client, Libgcrypt, libsmtp, etc. A partial list is maintained by the Free Software Foundation at: http://directory.fsf.org/license/LGPLv2.1/. However, this list is likely incomplete and you would have to do a full audit of your source code to get a comprehensive list.

The point is that whether your application uses the LGPL portions included with the commercially licensed Qt, or it just happens to use one of the hundreds of LGPL licensed software libraries, your corporation is already committed to abiding by the terms of that license. Consequently, there is only a minor reduction in liability in using the commercially licensed version of Qt with its substantial increase in license cost. The cost/benefit of that equation is something that is well worth reviewing.

## Summary

There are many good reasons to use the commercially licensed version of Qt and this paper has reexamined the five major ones in light of Nokia's divestiture of the commercial licensing and support business. In each of these areas – infringement indemnification, supplier relationships, static linking, safety requirements and LGPL risk reduction – Nokia's divestiture has changed some fundamental arguments for acquiring commercial licenses. Whether the change in these fundamentals is significant enough to reverse an existing decision to use the commercial version will depend on the individual risk adversity of a specific company along with the creativity of its engineers. However,

*Copyright © 2011 Integrated Computer Solutions, Inc. • www.ics.com • Phone: 617-621-0060*

we expect that for many companies, these changes are significant and that the number of companies using the commercial version of Qt will continue to drop precipitously as it has done since the first availability of an LGPL version of the software.

## About the Author

Mark Hatch is the Chief Operating Officer of Integrated Computer Solutions, Inc. (ICS), the largest supplier of Qt consulting and training services in North America. A 30+ year industry veteran, his career has spanned from the very early days of UNIX at the University of California, Berkeley in the 1970s to the exciting open source revolution of the present. His use of Linux dates back to 1993 where he had to insert over 20 floppies to install Linux Release 0.99 p13.

Mark joined ICS in 1995 as VP of Marketing and today manages the Qt business, both products and consulting services, at ICS. Previously, Mark was VP of Marketing for a networking middleware company as well as Manager of Software Marketing for Apollo Computers. In this latter position, he played a key role in the formation of a number of industry groups including the X Consortium and the Open Software Foundation. The X Window System and the Distributed Computing Environment (DCE) achieved their current position of industry standards in part because of the efforts of Mark and his marketing team at Apollo. Mark has a Masters in Electrical Engineering and Computer Science from the University of California, Berkeley. He also has a Masters of Business Administration from Boston University.

## About ICS

Integrated Computer Solutions, Inc. (ICS) is the leading expert in Qt, MeeGo, and GUI application development. As the largest independent Qt consulting firm in North America, ICS has extensive experience solving development challenges for all types of Qt applications, including MeeGo initiatives and applications leveraging the latest Qt functionality. ICS is also a Nokia, Qt-Certified Training and Consulting Partner and has provided training and consulting services to most of the Fortune 500, including many whose technology defines their core business. In addition, ICS sponsors the ICSNetwork, an online training center where users can learn advanced techniques for developing with Qt. More information can be found at www.ics.com.

*Copyright © 2011 Integrated Computer Solutions, Inc. • www.ics.com • Phone: 617-621-0060*