

Full Spectrum Software

*User interface design for linux, mobile and cross-platform products
by Andrew Dallas, CTO, Full Spectrum Software*

Qt Development

At Full Spectrum Software, we have been working on Windows, Mac, Linux and a variety of embedded and mobile environments for many years. One of the best tools for cross platform development has been Nokia's Qt development environment. Qt allows us to create one set of source code that will run on all three of our major platforms and most importantly will retain the familiar look and feel of both Windows and Macintosh. It helps us save our clients time and money if they need a professional looking application that runs on multiple platforms. It is our preferred development tool for Linux platforms. Qt is a very powerful platform, allowing for custom controls and rich graphics to be created when needed. In addition, it also supports several embedded platforms such as Windows CE, Windows Mobile and Symbian.

If Qt is so great, what's the problem?

The trend is that clients increasingly want elegant graphics and animations to more powerfully convey meaning, intent and workflow as well as dramatically improve ease of use. Graphic designers are engaged by the client to create these aesthetically pleasing, powerful graphical user interfaces.

Designers naturally use tools that they are comfortable working with such as Adobe Illustrator and Photoshop, and create JPEG files, or other graphic images to illustrate certain design elements.

The software engineering team must then gather and use all those graphical assets, into the user project in order to create the look and feel the client wants. In this workflow only the graphical assets are used and if the graphics need to be scaled or modified for reuse elsewhere in the application, the results may not be terribly crisp. This, by the way, is the same way an old fashioned Windows Forms based UI would be developed. This method is tried and true but has significant limitations.

This is a distinct two step process where the graphic designers develop form layouts and graphic design elements and the software engineers code the assets into the user interface. There is often some degree of iteration as the client requests certain amounts of re-work or changes to make the UI more precisely meet their needs. This can be an



[Click here to receive
a PDF portfolio
customized just for
you.](#)

Or

[Click here to email a
request for a
customized
portfolio.](#)



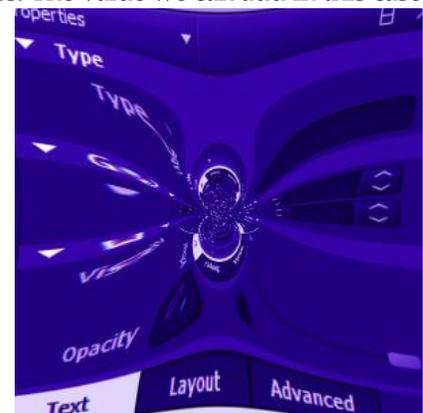
inefficient and time consuming process where both teams are working fairly independently.

The Competition's Solution

As illuminated in our online [WPF Webinar](#), with the advent of WPF, Microsoft has introduced a huge paradigm shift. Microsoft has created an environment that allows graphic designers to work with tools they are familiar with. For example, Expression Blend© is quite similar to the Adobe© design tools. Using these tools they can generate the page layouts and graphical elements in what is known as a declarative format, which in the case of WPF is a text file or more accurately an XML based file called XAML. The XAML file describes all of the layout and graphic elements but does not contain any of the programming logic. WPF allows the graphic designer to describe one set of custom controls, buttons as an example, and then apply that button style to as many buttons as they wish.

Now the software engineering team can re-use all the graphic elements created by the design team simply by coding the logic of the application. There are some caveats of course. The design team does have to understand how WPF works and how XAML files need to be organized. And in practical use, there is a level of rework software engineers should expect in organizing XAML libraries for best practices. However, this technology creates a highly efficient process where the design team can begin working with the software engineering team almost on day one and both teams are now working efficiently and collaboratively.

When using WPF our advice to our clients is to introduce our software engineering team to the design team as soon as they are engaged. We can save our clients time and money by becoming involved very early in the design process. The value we can add in this case relates to the aforementioned caveats that the design team does indeed know exactly how to use and organize the WPF design tools and architect the XAML files correctly. If this is not the case, our software engineering team needs to work with the design team on day one to assist in organizing the model to be employed. Otherwise, utilizing WPF incorrectly is just as inefficient as using any other professional software tool incorrectly. It wastes the client's time, effort and money.



The Solution for Qt

Nokia has observed the success of WPF and Qt is not standing still. Nokia presumably understands how popular and efficient WPF is and they are furiously working on a new release of Qt dubbed Qt Quick. Qt Quick takes an approach somewhat similar to WPF.

Nokia describes it this way: "User interfaces and their behavior are described using QML, an extension to JavaScript that lets developers and designers use a declarative syntax to specify each user interface in terms of QML elements. These elements are a sophisticated set of graphical and behavioral building blocks that can be combined



Full Spectrum Software

together in QML documents to build components ranging in complexity from simple buttons and sliders, to complete Internet-enabled applications. To allow the implementation of more advanced behavior, QML integrates tightly with imperative JavaScript code. The JavaScript environment provided by QML is stricter than that in a web browser.”

Notionally Qt Quick will be just as efficient and just as powerful as WPF but its major competitive advantage is that, unlike WPF, it is still a multi-platform tool.

The Qt Quick Scoreboard

Based on our review of the Release Candidate, Qt Quick is a very powerful environment and if it performs as well as WPF, it will be a very valuable tool in the multi-platform arsenal. Qt Quick’s downside is that Nokia is clearly playing a game of catch up with Microsoft. WPF was released much earlier and has the benefit of a highly experienced user base and the incremental improvements that inevitably flow from that. Without belaboring the obvious, WPF is just one more tool in Microsoft’s formidable collection of development tools.

However, Qt Quick has several competitive advantages that are not immediately obvious. Qt has long enjoyed the support of the Linux community, a relatively large and growing faction within the scientific software development community. Linux developers are particularly passionate about their platform and tool sets. If it performs as

```
if (display.currentOperation === "Add") {
  display.text = Number(display.currentValue) + Number(display.currentValue);
} else if (display.currentOperation === "Subtract") {
  display.text = Number(display.currentValue) - Number(display.currentValue);
} else if (display.currentOperation === "Multiply") {
  display.text = Number(display.currentValue) * Number(display.currentValue);
} else if (display.currentOperation === "Divide") {
  display.text = Number(display.currentValue) / Number(display.currentValue);
}

if (op === "+" || op === "-") {
  display.currentOperation = op;
  curVal = display.text - val;
  return;
}
```

advertised, it should be every bit as powerful and efficient as WPF. For Linux programmers, Qt Quick looks to be the same game changer that WPF was for Windows programmers.

In addition, Apple is continuing to court the scientific and medical device market and their professional grade hardware platforms are every bit as powerful as multi-core Intel machines. If Apple makes continued inroads into the scientific software community, there will be the inevitable effect of adding further support for Qt Quick.

When will Qt Quick be available? As of this writing it is available as a Release Candidate. The original release date was the end of 2010. So to borrow an expression from the software development community, it should be ready “any day now” by which we mean sometime in 2011. What to do now? Early adopters should start to learn the product. Many engineers are challenged by the migration to WPF and any new tool will cause growing pains. The first few revisions of Qt Quick will likely have the usual contingent of bugs and limitations and knowing those will help you to produce commercial quality products in the short term.



Full Spectrum Software

About Full Spectrum Software

Full Spectrum Software is an ISO 13485 certified, 15 year old consulting firm specializing in the development of embedded and applications software for the medical, life sciences and scientific industries. The company has delivered over 100 commercial products to market.



About the Author

Andrew Dallas, the firm's CTO, is widely considered one of the leading authorities on best practices in FDA and ISO 13485 controlled software projects. Andrew serves on the Editorial Advisory Board of Medical Device and Diagnostic Industry magazine and he has published extensively in major trade and peer reviewed technical publications.

Contact Ken Carson, ClientServices@FullSpectrumSoftware.com
(508) 620-6400 • 225 Turnpike Road • Southborough, MA 01772 •
www.FullSpectrumSoftware.com