MFC to Qt

MIGRATION SERVICE

KDAB

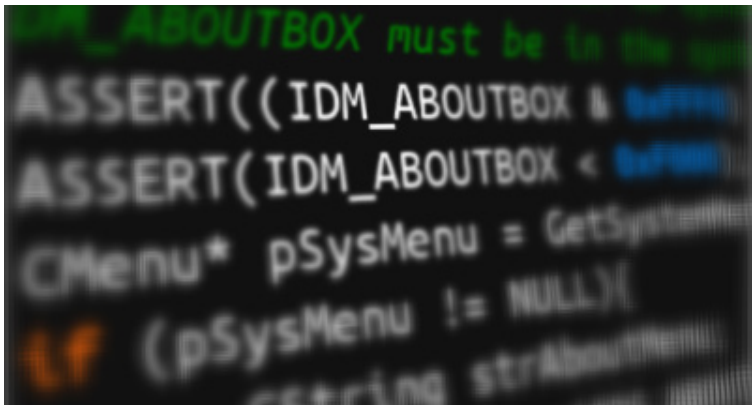
If your project relies on MFC, chances are you're trapped by a framework on life support, unable to take advantage of productivity gains, new libraries, multi-display hardware, cross-platform support, mobile app development – the list goes on. What's more, it's getting harder to find developers who want to work on UX technology that's a career dead-end.

It can be exceedingly painful to move away from something so deeply integrated into your codebase – especially for large or complex projects – so you stay on a legacy framework far longer than you anticipated. With strapped resources, limited budgets and shifting priorities, the right moment to move off a legacy framework never comes. Yet the pressure to migrate to a newer framework mounts week by week. Let us help.

Staying on a legacy framework keeps your talent pool small and dwindling, making it difficult to stay competitive.

Through porting many MFC projects, KDAB has accumulated a vast wealth of knowledge on how to migrate systems still under active development with minimal disruption.

As the leading C++ cross-platform framework, Qt is a natural candidate for MFC replacement as it can handle embedded systems, desktop development, and mobile platforms. While Qt and MFC are both over 25 years old, that's where the similarity ends. Qt is actively maintained by a vibrant developer community, upgraded regularly to embrace the latest programming improvements, and expanded to cover new devices and operating systems. Using QML's declarative UI building greatly simplifies adding new dialogs and UI elements.



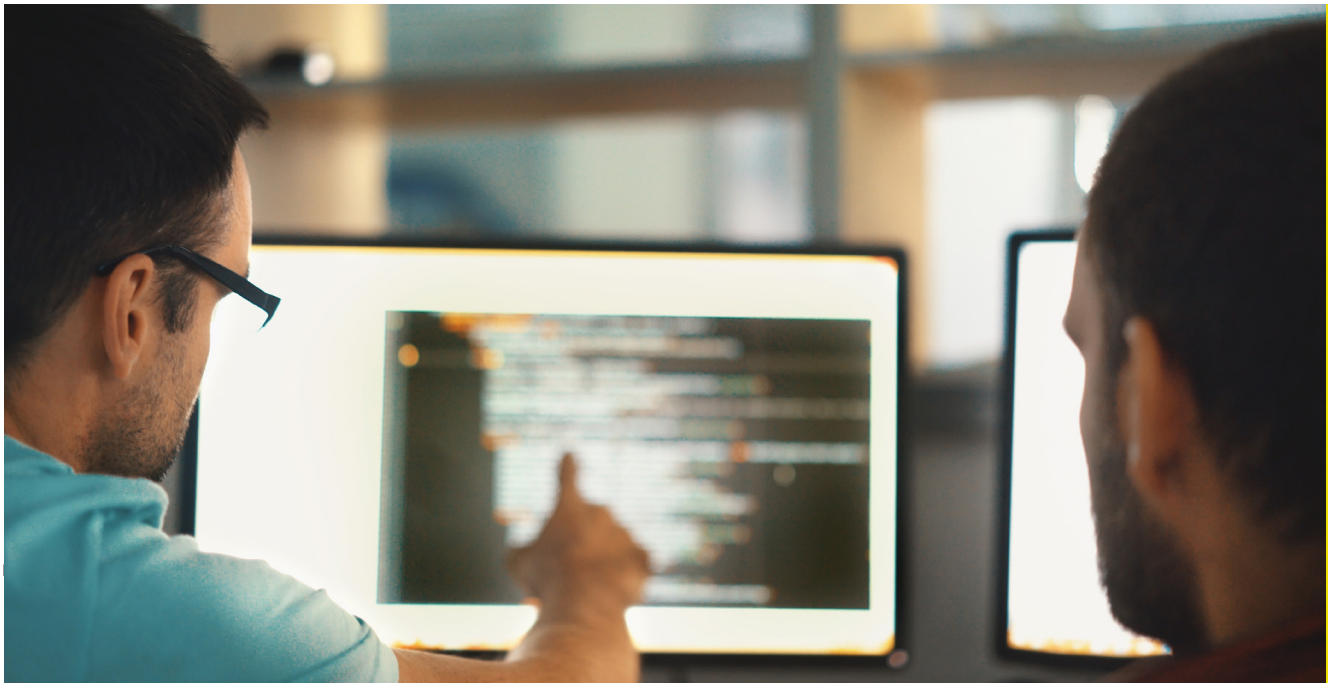
A migration from MFC is a one-time activity, which means you'd be spending a lot of your concentrated engineering time on something you'll never need again.

We're happy to save you the pain of trial-by-error learning and let your engineers do what they do best – expand your brand.

WHAT MAKES FOR A SUCCESSFUL MIGRATION?

A clean MFC migration requires expertise and tooling in specific subsystems:

- **User interface** – moving dialogs and UI elements to Qt equivalents. This may optionally include an expert assessment on UI usability and design, and making any UI changes as a result
- **Build system** – moving from Microsoft-specific builds to a platform-independent make system
- **Non-UX code** – moving the core MFC classes (like strings, files, networking, etc) to pure C++ or C++ with Qt as desired
- **Windows APIs** – removing any Windows-specific APIs and replacing them with OS-generic equivalents



At KDAB, we've helped many companies migrate from MFC to take advantage of Qt, giving their products a renewed vitality. Migration can be daunting and intrusive, but with our migration expertise we transform that scary process into one that is calm, controlled, and replicable. We've also perfected several techniques that allow us to gradually move an MFC application to a Qt/MFC hybrid. This lets you still run, fix, and augment the program while you're migrating your software and training your staff, greatly helping to relieve the stress of a porting effort. We also have tools that we've created specifically to help address MFC migration – to help ease the run-time environment changes, as well as automate some steps of the migration process.

For any large MFC port, we recommend an initial pilot project that consists of migrating a small subset of your application's code to Qt. This allows us to fully understand your software, gives us the ability to better estimate the scope and time of the overall project, and gives you confidence that the program is solid before the entire project is completed.

SIX STEPS FOR A CLEAN MFC MIGRATION

1. **Evaluation**: We start with a free migration evaluation to determine the scope, timeline, complexities, and cost, ensuring that the porting process meets both needs and expectations

2. **Discussion and design**: We discuss any specific coding standards and procedural guidelines particular to your organization and design the port with these in mind, as well as create a post-port plan for refactoring, rearchitecting, or making future UX changes

3. **Tooling**: We port over some of the legacy code using a proprietary MFC-to-Qt transformation tool that automates a portion of the mindless work to save on overall development cost

4. **Porting**: We hand off the remainder of the port to Qt and MFC experts, who migrate the rest of the software and collaborate with your engineering staff as desired

5. **Testing**: We test the port against our own test scaffolding as well as any additionally required testing

6. **Training**: We hold specialized workshops and/or training as needed to ramp up your engineering staff on the new Qt frameworks and tools

Many mature projects have user interfaces that are significantly dated so we often recommend a usability assessment. This can help identify new user interaction paradigms, areas of possible confusion, and places for workflow streamlining. To avoid destabilizing your software, we actually advise against making UI changes during a migration – it's significantly better and less risky to create a controlled baseline first. However, knowing which UX elements will change after migration can help lessen testing efforts and identify appropriate structural changes.

Migrationwapitalize on the core value you bring to your product. Considering the number of hours it takes engineers to learn how to move a product from MFC to Qt 5, using KDAB is a great way to save on time, money, and headaches.

Moving to Qt will open up your product to the benefits of a modern, actively supported, comprehensive framework backed by a huge community:

- Qt is continually updated through a regular release cadence with feature enhancements and bug fixes
- Qt uses modern development method-ologies and embedded-friendly C++, supporting C++11 features (C++14 coming soon)
- Qt 3D provides full 3D support with optional physics-based rendering (PBR)
- Qt has components for graphics, IoT devices, Bluetooth, Sensors, and other peripherals
- Qt supports comprehensive multimedia standards (including DRM)
- Qt Quick and QML provide a powerful but simple scriptable UX
- Qt Creator IDE allows closer collaboration between designers and developers
- Qt offers multi-platform support and portability to other operating systems like Linux, Windows, Windows Embedded, and Android

About the KDAB Group

The KDAB Group is the world's leading software consultancy for architecture, development and design of Qt, C++ and OpenGL applications across desktop, embedded and mobile platforms. KDAB is the biggest independent contributor to Qt. Our experts build run-times, mix native and web technologies, and solve hardware stack performance issues and porting problems for hundreds of customers, many among the Fortune 500. KDAB's tools and extensive experience in creating, debugging, profiling and porting complex applications help developers worldwide to deliver successful projects. KDAB's trainers, all full-time developers, provide market leading, hands-on, training for Qt, OpenGL and modern C++ in multiple languages. Founded in 1999, KDAB has offices throughout North America and Europe.

KDAB

www.kdab.com