The Perfect Framework

for Industrial Applications

Introduction

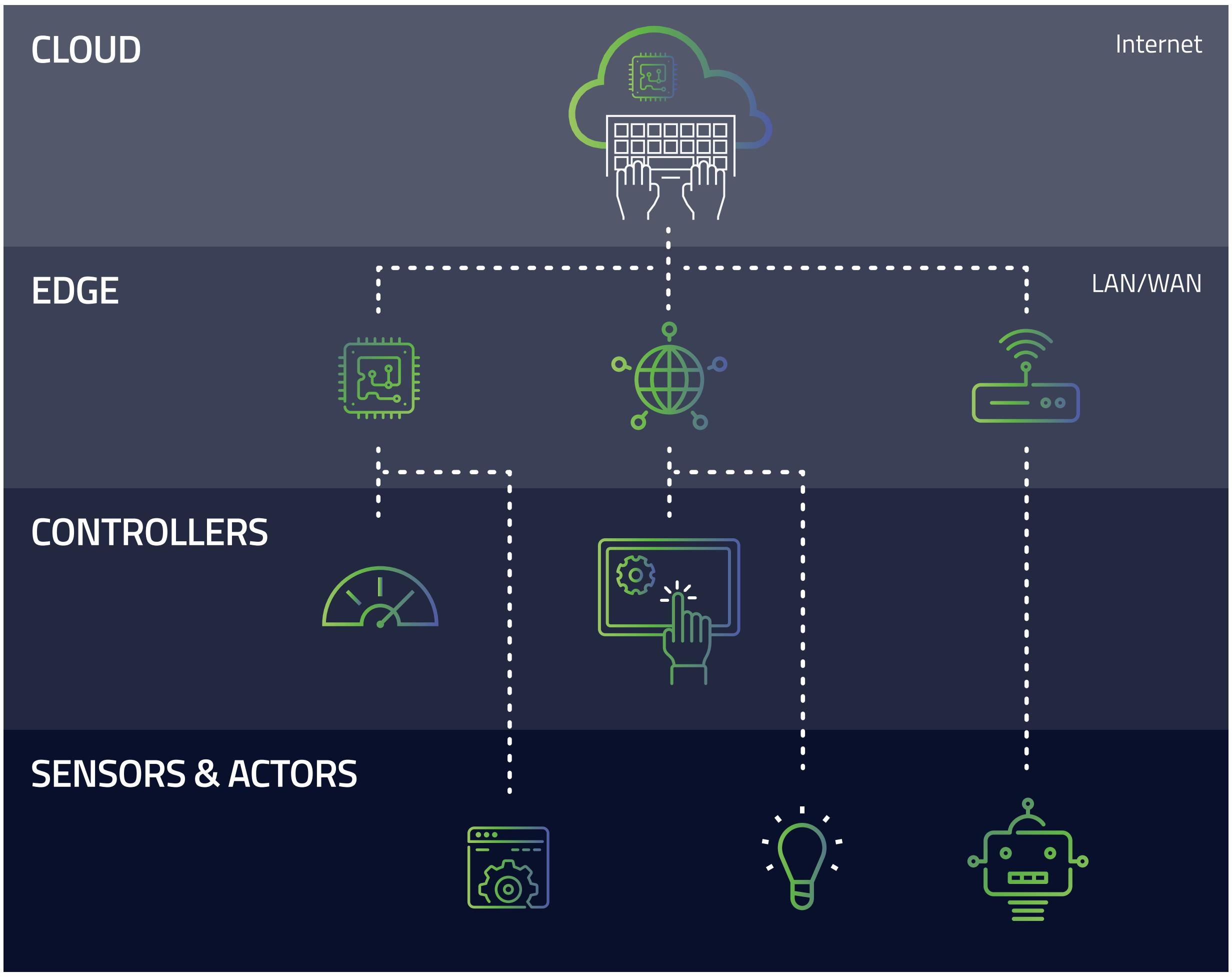
Qt has become a strong framework for creating industrial applications and is one of the most dependable choices in this segment. For nearly two decades, manufacturers have consistently depended on Qt to produce increasingly complex systems that maximize production uptime and product reliability. But requirements for the industrial market now include streamlined design workflows, multiple target deployments, remote browser accessibility, connectivity through new protocols, and many other demands. These are all areas where Qt continues to excel and, with every new release of the framework, new capabilities are added to address industrial concerns.

In this eBook, we look at the latest trends in the industrial market and explain why we think Qt helps companies tackle those trends. We also examine several real-life use cases where people have successfully used Qt in their companies to meet business goals. When you finish with this eBook, if you're interested in more specifics about how Qt solves industrial challenges, you may want to read our eBook entitled, "The Nuts and Bolts of Qt Industrial Applications".

What’s an industrial application?

Qt applications have commonalities that cut across many fields so we’ve adopted a relatively broad definition for this e-book. We consider an “industrial application” to be one that is:

- Used to install, set-up, optimize, or operate equipment
- Deployed on an embedded system, desktop application, wearable, mobile/tablet, gateway, headless device, or any combination thereof
- Positioned as an edge device or manages edge devices in the Internet of Things (IoT)
- Demanding significant reliability and resiliency with high availability and functional safety requirements – including possible adherence to safety standards like IEC61508, IEC62304, or ISO26262.



This definition includes conventional industrial disciplines like SCADA, process control, and remote operation using traditional hardware like PLCs or CNC machines. But it also includes tablet-driven companion apps to configure sound systems, consumer-operated electric-vehicle charging stations, industrial gateways, and inventory-warehouse wearable wristbands.

Most applications in markets from industrial through infrastructure have similar software architectures. This fact allows Qt to be applicable in fields as diverse as electronics, marine, pharmaceuticals, building automation, warehousing, agriculture, smart cities, facility management, power generation/distribution, HVAC, robotics, or manufacturing systems.

Why is Qt ideal?

Qt is a cross-platform development framework that can run on various operating systems as well as software and hardware platforms with little or no change in the underlying codebase, while still being a native application with native capabilities and speed. And that's just for starters; there are many reasons why Qt is an ideal framework for industrial applications:

- One of the biggest developer communities of any user interface framework, which means tons of relevant examples, software libraries, engineers trained on how to use it, and companies supporting it
- Very active contributors and support for the latest CPUs, boards, graphics, peripherals, and protocols
- Support for multiple display-technology options including widget-based, declarative, 3D, or a mix – allowing developers to build a complete UX within one tool
- Powerful integrated development environment (IDE) that is fully Qt-aware, making it easy for developers to create, analyze, debug, and optimize applications
- Support for many protocols that cover the majority of different automation-segment applications (manufacturing, building automation, Industrial IoT)
- Inherent cross-platform support for multiple boards and operating systems, as well as for a mix of embedded, desktop, and mobile devices

When it comes to cross-platform development, aren't tools like HTML5 also applicable? There are a host of technical reasons that make Qt superior to HTML5 for industrial applications. The Qt framework is built upon decades of robust development and as a result is more reliable and stable. With native C++ performance, Qt provides smooth user interfaces and consistent machine control. And Qt applications can be made much more secure than web apps. We address these features in more detail in a separate eBook entitled, "The Nuts and Bolts of Qt Industrial Applications."

How Qt became a major player

The same factors affecting the industrial market are driving software around the world:

- ✓ Ever-increasing application complexity
- ✓ New UI demands for richness, animation, and ease-of-use
- ✓ Internationalization and personalization
- ✓ High performance across multiple diverse platforms
- ✓ Functional safety and high reliability
- ✓ Requirement to meet all needs with a common platform

Because Qt is used by many customers in many markets, it has benefitted from best practices and lessons learned over two decades through a worldwide demand for improvement from many sectors.

Real-world example: Parker Hannifin

Parker Hannifin is a leading manufacturer of motion and control technologies and systems, providing precision-engineered solutions for mobile machines addressing a wide variety of markets, such as agriculture, construction, forestry, material handling, and transportation. The company sells directly to heavy equipment OEMs who in turn integrate Parker Hannifin electronic control systems and instrumentation displays into their trucks, cranes, buses, and tractors. The company discovered that their customers were struggling with the software complexity needed to integrate their equipment into professional looking, in-cabin displays. Moreover, because the system required a proliferation of displays for several disparate systems, it was not only more difficult for users to operate, it was more expensive to manufacture as well.

To help their customers create more user-friendly interfaces and bring their mobile machines to market quicker, Parker Hannifin decided to create a tool that would allow OEMs to simply and quickly assemble custom applications from a large assortment of pre-built and tested components. This tool needed to create software that could interface with multiple pieces of equipment on a truck or tractor through a single touch-screen display using a unified user interface. There was also a clear requirement for the tool to be simple and easy to work with. They decided to use Qt, as it could create polished user interfaces with lots of customizability, it had a powerful high-level declarative scripting language (QML) that would be easy for the OEMs to use, and it could easily access underlying hardware such as the J1939 bus.

"The wow-effect was there when you program with QML. It keeps on astonishing with it's ease-of-use and efficiency. I got that feeling right in the beginning and it has even amplified along the way. QML has provided a solid base for our software."

Tommi Forsman – Principal Engineer, Parker Hannifin

Parker Hannifin built their product, the Parker Application Designer, with a combination of Boot to Qt, the Qt Creator, and Qt consulting services. They directly incorporated the Qt Creator into their tool, leveraging the power and capability already built into the Qt IDE. They added a library of applications tailored for different equipment options, and the ability to offer several style sheets and themes for a custom look and feel. Providing an application-based platform let a single instrumentation display control the individual machines' functions, making it much easier for users and less complex and expensive to produce, as well as giving Parker Hannifin the advantage of using a single module for all software updates, debugging, and development. The end result is OEMs are now able to put together complicated instrumentation displays with a drag and drop interface and simple high-level QML scripting – without needing to delve into complex programming.

Real-world example: DMG Mori

As ”Global One Company” DMG MORI is a leading producer worldwide of machine tools. The range of products includes turning and milling machines, as well as Advanced Technologies, such as ULTRASONIC, LASERTEC and ADDITIVE MANUFACTURING, as well as automation and complete technology solutions. DMG MORI bundling its technology excellence in the leading industries “Aerospace” “Automotive”, “Die & Mold” as well as “Medical”. The “Industrial Services” offers a wide range of customer-oriented services such as training, repair, maintenance and spare part services covering the entire machine life cycle. The APP-based control and operating software CELOS and the exclusive DMG MORI technology cycles and Powertools enables DMG MORI to shape Industry 4.0. 10,000 DMG MORI machines have already been fitted with CELOS and are in the market.

DMG Mori needed to streamline a complex workflow for operators, technicians, and managers as well as introduce a completely digital and paper-free process.

To accomplish their goals, DMG Mori used Qt to build an app-based user interface, named CELOS, for all of their equipment. They were able to improve usability – even for highly trained operators – by making the displays easy to use and interpret through an attractive user interface. Relying on the speed and reliability of C++, they were able to process and handle data in real-time without delays, critical for controlling precision milling equipment. With Qt’s cross-platform strength, they were able to use a single development team to create the embedded version of the product as well as a PC version.

Software trends: What's happening in industrial?

Some people think software developed for industrial applications hasn't changed much in the last couple decades. While it's true that this space is a bit more resilient to the whims of consumer desire, a proliferation of new requirements include support for new protocols, needs for visualization and charts, customer demand for internationalization, touch screens and virtual keyboards, easily interpreted dashboards, and remote browser access. These features are all driven by numerous market trends that affect the look and feel of industrial applications and how they're designed and built.

Mobile Influence

The pocket convenience that Apple iPhone and Google Android smartphones provide has made a permanent impact on customer expectations about ease of use, functionality, and immediacy. Daily interactions on a mobile device change a person's worldview of what an acceptable user interface should look like – even for industrial applications.

To that end, Qt helps modernize applications with touchscreens, gestures, and haptic controls, helps support features like app stores, and helps employ focus-directing animations. With Qt, developers are able to build a responsive and beautiful HMI that users will want to interact with – just like their smartphones.

Device integration and remote access

While people increasingly expect their industrial equipment to sport a smartphone-like interface, there are many manufacturers creating smartphone/tablet apps that interface with their equipment. Other manufacturers are having their equipment host a "responsive" web interface that can be used easily on a desktop or mobile device – without requiring app installation. These common capabilities lead people to expect that they can use their personal devices to communicate with most industrial systems; giving a piece of industrial equipment that capability makes it seem more modern, approachable, and personalized.

Qt is well known for its ability to create cross-platform applications. It's easy to create the primary interface for an embedded system while using the same code base to develop a mobile companion app. That dual purpose not only saves time in creating a single set of common libraries but also lends itself to a shared look and feel. If you choose to go the "app-less" device-hosted website route, it's also easy to bundle a remote browser-capable interface into your industrial app with QtWebGL or Qt-compatible WebAssembly.

Devices and industrial applications

When does it make sense to create a companion app or remote-control web access for your hardware? Here are a few reasons why manufacturers have tied their industrial application to personal devices:

- Inconveniently accessed displays – thermostats, HVAC controls, or alarm systems
- Complex setups and configurations – concert-hall audio systems, laboratory automation, or CNC machines
- Primitive or non-existent displays – gateways, pumps, or roadside sensors
- Sophisticated diagnostic reporting requirements – automotive systems, power plants, or production lines
- Mobile viewing requirements – medical equipment, security systems, or hazardous work robots

Time-to-market pressure

Consumer electronics and mobile apps can have rapid release cycles of weeks or even days between updates. That makes customers far less tolerant of waiting six months or a year for the next release of your system that fixes important bugs or adds critical features. These pressures not only tend to shorten development cycles they also imply a need for over-the-air (OTA) software updates.

With Qt Quick and QML, Qt is well suited to creating rapid prototypes. And while there's no silver bullet for maintaining quality while cutting development cycles, Qt has a powerfully expressive framework that makes it easy for developers to create, test, and debug their code in record time.

Another significant part of reducing development cycles is in improving the workflow between designers and developers. With Qt Design Studio and Qt 3D Studio, user interface designers can build an HMI using powerful visual tools, making those assets directly available to developers. That alleviates the sometimes-painful round-trip between designers and developers caused by dissimilar, incompatible tools.

COTS / open source

It often doesn't make sense for developers to reinvent the wheel every time they need a new capability – this significantly adds to the development cost of a new application and introduces bugs and inefficiencies that others have already fixed. Which is why nearly every project liberally incorporates other libraries, components, and applications that are either commercial off-the-shelf (COTS) or open source.

Nearly every open source library – regardless of language – provides a C/C++ API, which allows Qt to easily include and interface with it. To help manage the complexity of licensing when open source and proprietary code are mixed, the Qt framework is available with multiple software licenses and is pre-integrated with many common license compliancy tools.

Cloud computing

Moving functionality into the cloud is a persistent theme in industrial applications and customers are asking for it primarily for two reasons. The first is that it allows bug fixes and feature updates for all deployed hardware simultaneously – updating the cloud server let's you automatically update every product in the field without your customer needing to organize a massive IT roll-out. The second reason is that putting data into the cloud allows that data to be shared across all customer sites and installations, dramatically simplifying synchronization and duplication issues.

Qt has implementations available for accessing most types of cloud services or data, such as REST and SOAP/WSDL APIs. Those that aren't directly available in Qt will almost certainly have a C/C++ API, allowing them to easily be integrated.

Data visualization

Many industrial systems generate a vast amount of data and analyzing that data can be difficult, tedious, or require special training. As a result, many systems are increasingly relying on data visualization to provide alternative ways of detecting problems, identifying patterns, or highlighting inefficiencies – especially when concerning predictive maintenance.

Qt provides a library dedicated to visualizing data in 3D with bar charts, scatter diagrams, and surface meshes as standard. A large number of samples shows how to use the data visualization library in a multitude of creative ways.

Cloud-based industrial is for everyone

The benefits of cloud-based industrial apps are not just for the customer. Continual access to customer data provides manufacturers with a number of key benefits:

- Insights into how applications are being used in order to improve products and anticipate customer needs
- New products based on anonymously collected data or its analysis
- Data-based services – such as predictive maintenance or reporting – that can be progressively expanded or monetized

Platform freedom

Desktops come in three flavours – Windows, Mac, and Linux. Mobiles are either iOS or Android. Embedded hardware is as numerous as grains of sand on the beach. Any way you look at it, it's suicide to lock yourself into a single hardware or OS platform today. Creating flexible software that's independent of platform and requires little change to migrate is the path with the least risk and most benefit.

Qt excels at creating cross-platform code – one of Qt's biggest claims to fame. Qt supports all desktop and mobile variants equally and it's usually the first graphical framework enabled on any new system-on-chip (SOC) or graphics processing unit (GPU).

Bill of materials (BOM) cost

No matter how specialized the discipline, cost concerns will never truly go away. However, global markets continue to shrink the planet, letting prices of products from low cost countries be seen alongside traditionally higher-margin geographies. This means there will be continued pressure to contain and lower costs to compete in a global market – even if the product has otherwise uniquely differentiating features.

Using Qt can't magically reduce your bill of materials. Or can it? Actually, Qt compares extremely favourably to other frameworks in terms of creating optimally performing code and minimally sized data. What this means is using Qt could in fact allow you to use a lesser powered CPU, fewer RAM chips, or a smaller flash disk than if your application was created using a more wasteful framework.

Developer ease

Developers are being asked to deliver more than ever before, faster than before, and of course with fewer bugs. Efficiency is at a premium.

Qt code doesn't write itself. But because Qt is expressively powerful, it allows developers to program with concise statements, saving time and brainpower.

While the industrial market is fragmented into thousands of different application areas, it's safe to say that the overall focus on automation has grown dramatically in recent years as have the demands for remote accessibility, connectivity, consumer-grade interfaces, and the like. A cross-platform application framework such as Qt clearly addresses many of this market's trends and challenges while being a relatively easy language to learn and progress. If you'd like more specifics on the industrial challenges Qt addresses, read our eBook entitled, "The Nuts and Bolts of Qt Industrial Applications".

Developer scarcity?
If you're worried about finding developers familiar with Qt, don't be: More than a million developers in more than 70 industries currently use Qt.

The Qt Company