Modernizing SCADA HMIs

Industrial Applications, iPhone Expectations

Christoph Sterz

Supervisory control and data acquisition (SCADA) systems have been around since the 1950's, far longer than most other types of computer applications. Their rock-solid performance has been responsible for the streamlining of any industry that needs precise and consistent controls: building automation, energy management, part machining, printing and packaging, robotic assembly, ship building, water treatment, woodworking, and many more. However, this long legacy can also carry a hidden drawback – the user interfaces of many SCADA devices look more appropriate as part of Windows for Workgroups than the modern age.

This situation is ripe for change. Now that everyone carries superior user interfaces in their pocket at all times, even the non-designers responsible for running the system expect their SCADA human-machine interface (HMIs) to have a certain level of polish and sophistication. Having implemented attractive SCADA HMIs for our customers, we believe that Qt is the right tool to build the modern SCADA system – here's why.

KDAB

This whitepaper provides a quick overview on why KDAB finds Qt to be an ideal tool for building today's modern SCADA systems.

Qt allows developers to build components that can run on embedded, desktop, and mobile computers

Close to the metal

SCADA systems interface to equipment that often has hard real-time requirements. In these cases, responding slightly late may be just as bad as not responding at all. You want the switch that flips the electrical grid to happen exactly when you tell it to – not after a brownout or an overload. With Qt's C/C++ backbone, developers can get maximum performance from their SCADA applications by creating code that talks directly to the hardware – no virtual environments, garbage collection, or unanticipated events happening in between.

The building blocks of any SCADA system are programmable logic controllers (PLCs), industrial computers that have a large number of I/O ports that are used to scan gauges or control actuators. The typical PLC implementation uses direct memory mapping to access its connected hardware, meaning you must be able to read and write to specific memory addresses. This is easy for C/C++ (and thus Qt) but often not available in other languages.

Deployable everywhere

Important to any modern SCADA system is flexibility – the HMI needs to be accessible from the embedded system, remote desktops, and roving mobile devices – something that KDAB does for customers often.

On the embedded side, you need a platform that can run on PLC hardware. While not every PLC is capable of running a graphical interface directly on its hardware, those that are, typically run a real-time operating system like VxWorks, QNX, WinRT, INTEGRITY, and RTLinux. Qt supports these operating systems, making SCADA systems as simple to support as any other embedded system. SCADA HMIs are also found on the computers monitoring the systems in the back office, which run standard desktop operating systems such as Windows, Linux, or macOS. They're also found on phones and tablets due to the increased need to support mobile trouble-shooters, supervisors, and headless machines. Of course SCADA apps on mobile devices must run on iOS, Android, and/or Windows 10, which is simple with Qt's cross-platform support.

Thankfully, Qt is a very broadly used and actively supported platform that runs on all these operating systems and is easily adapted to others. Perhaps most importantly though, all of these environments are supported from the same code base, allowing developers to build components that can run on embedded, desktop, and mobile computers.

Qt supports all of today's UI paradigms, controls, and behaviors, making it easy to design really attractive HMIs that users intuitively understand.

Having people physically present at every machine is extremely inefficient, which is why Qt has a number of handy tools for remoting

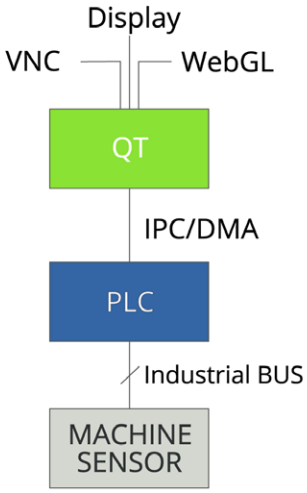
Modern HMI

To make a modern SCADA system, you need to incorporate modern user interface elements; the arcane keyboard commands, 2D sprites, and 8-bit graphics of yesteryear aren't enough. All of today's expected user interface paradigms, controls, and behaviours – like drag and drop; spinners, sliders, and splitters; touch, pinch, and zoom; tables and popups; and much more – are part of Qt. That makes it easy to design really attractive HMIs with controls and interactions that users will intuitively understand. It's also easy to customize the appearance of the HMI for either customer-specific branding or white-labeled products – a nice detail that enables your products to be tailored for each customer engagement. This is all relatively straightforward with Qt's widget styling and easily modified Qt Quick screens, features that let developers make global style changes to applications in a few highly localized classes that will percolate throughout the HMI. KDAB has helped many clients take their UI from dated to dazzling using all the tools that Qt provides.

Many SCADA interfaces require three-dimensional models of the plant floor, building, or equipment being automated. That's achievable with Qt 3D, an entire set of Qt classes created and maintained by KDAB, that lets you build 3D objects and can take advantage of GPU hardware-accelerated graphics. Developers are able to directly import any necessary 3D models using assimp, the Open Asset Import Library. Another option is to use QOpenGLWindow, which allows developers to use OpenGL code more directly – a great option if there's already OpenGL code that does what's required. Our recently released KUESA™ for Qt 3D tool is also an extremely handy option for importing and manipulating 3D assets in a SCADA system.

Video is used in SCADA to check remote locations and equipment, which allows a supervisor to monitor situations that may be dangerous or impractical for a person to be physically present. Whether it's making sure the cooling system for a nuclear power plant is operating properly, verifying a robot's painting job for parts on an assembly line, or detecting a quality inspection machine that accepts incorrect products – remote cameras are a necessary part of many SCADA systems. Thankfully, Qt has APIs for embedding video streams in individual windows, letting a single HMI provide oversight of an entire operation instead of requiring dedicated video monitors.

Qt Safe Renderer can be certified under ISO 26262 for automotive, EN 50128 for railway, IEC 62304 for medical, or IEC 61508 for generic industrial systems.



Qt can provide multiple channels to export data views to external agents

Remote observation and control

Video isn't the only important remote feature of a SCADA system. Even more important is the ability to remotely run the HMI. In these situations, the hardware supplies the data that is visualized on a remote tablet or desktop to allow operators from across the floor – or across the country – to keep tabs on their equipment. While this is critical for PLC systems that have no displays, requiring people to be physically present at all machines with displays is extremely inefficient. Any modern factory expects their SCADA equipment to have remote views and operation.

Qt has a number of tools that are very handy for remoting. Qt VNC provides a remote desktop using the standardized VNC protocol for perhaps the simplest solution. However, newer additions to the Qt family include WebGL for transporting 3D imagery and WebASM for speedy browser run code that allow even more customized remoting solutions. For developers that want to link their back-end PLC to a unique front-end application, QRemoteObjects allows them to seamlessly share logical state information between the two machines. Many more approaches are possible – at KDAB, we've used the flexibility of Qt and C++ to create several custom remoting options besides these.

Stable, safe, and secure

Another critical attribute of SCADA systems is their longevity. SCADA equipment may be deployed in the field for many years, even decades. It cannot afford to rely on tech fads or moving targets. Qt has been around a long time and is built on reliable, proven technology. Long term support is available for select Qt versions, ensuring that those platforms will be stable, supported and maintained for many years to come. Companies using Qt support can decide on what patches or relevant updates they wish to incorporate, keeping products relevant for a very long time. At KDAB we support clients on both old and new Qt versions. We excel at advising clients when to move to a newer version and helping them modernize in order to preserve their investment.

Depending on their industry, SCADA systems may fall under ISO or IEC certification requirements. Qt Safe Renderer, a component provided under Qt license, can assist with certification. A safety-critical system that uses the Qt Safe Render to implement its graphical warnings can be certified under ISO 26262 for automotive, EN 50128 for railway, IEC 62304 for medical, or IEC 61508 for generic industrial systems.

Qt uses the latest SSL and TLS implementations to safely encrypt data communications for cybersecure remote access and control.

As SCADA is inherently dependent on remote control, security is an obvious necessity. Because Qt incorporates the latest SSL and TLS implementations, data communications can be encrypted with the latest, safest, libraries. This enables SCADA systems with Qt to transfer data, HMI images, or control information securely.

Qt for SCADA

Here we've listed many of the reasons that Qt makes an excellent tool for those building SCADA systems, especially those who want to modernize their equipment's HMI to reflect a more modern ascetic. Hopefully readers who are building the next generation of SCADA system will bring us HMIs that will make iPhone users take notice.

If you have any questions about modernizing your SCADA system that you'd like to discuss with an expert, please feel free to contact us at scada-hmi@kdab.com.

Christoph Sterz

Christoph is a software developer at KDAB who has been developing with C++ for over 15 years and Qt for the last seven. A contributor to GammaRay, KDAB's in-depth Qt-analysis tool, Christoph specializes in performance optimization for embedded systems – from industrial automation to automotive infotainment systems – and has advised large software teams on Qt/QML. Christoph holds an MS in IT-systems Engineering with distinction and is a published author in several areas including human-computer interaction, distributed systems, and IT security.

www.kdab.com

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 and is one of the biggest independent contributors 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, Europe, and Asia.