# Case Study

## Weather Station Interface with XML and Graphic Data from the Internet

What started out as a quick project for one of our new engineers (Jeff) to learn Qlarity, quickly turned into a classic example of "feature creep."

Located in the QSI lunch room is the display for a weather station (Vantage Pro™ from Davis) that is mounted on the roof of our building. Curious employees can punch the buttons on the display in an effort to figure out the wealth of information the system provides.

Realizing everybody was really only curious about a few key indicators Jeff used one of our Qlarity-based terminals, the QTERM-G70, to summarize the information provided by the weather station in a simple graphical format (Figure 1).

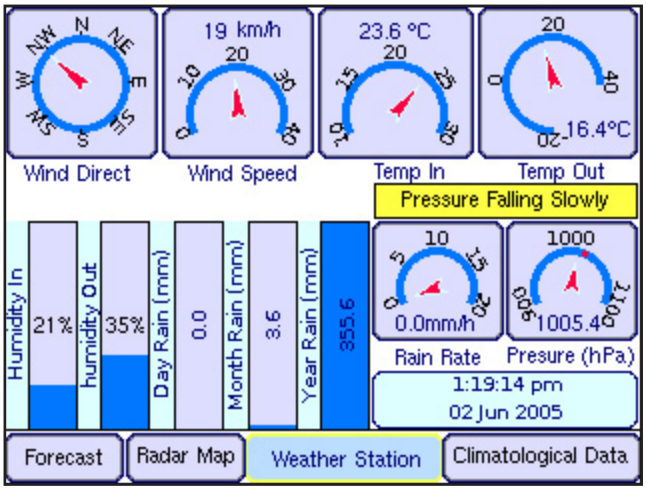


*Figure 1 - Weather Station Data on QTERM-G70*

Jeff connected the weather display to the QTERM-G70 using a serial link. He soon realized the protocol for the weather display was a variant of an EIA-232 serial protocol and modified the existing Qlarity serial object to communicate with the weather display (Figure 2).

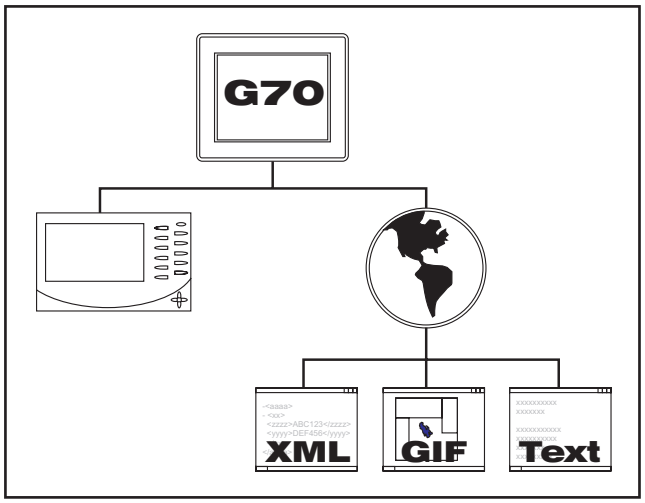


*Figure 2 - Weather Station Architecture*

Once the connection was made Jeff had access to a wealth of meteorological data that was captured and logged by the station. He used the existing Qlarity objects (gauges, text, graphics, etc.) to display temperature, humidity, barometric pressure, rain fall, etc., on the terminal.

Realizing employees really wanted the five-day forecast, similar to what you see on the local news, Jeff began to search the Internet for this information. He found three sites with XML and graphic data providing the information he needed to complete the project.

Jeff found XML weather forecasts for Salt Lake City area from the National Weather Service and a GIF radar map image which was regularly updated throughout the day.

The Qlarity language is very powerful, so Jeff created an object to convert the GIF image to the image format supported by Qlarity (Figure 3) and a simple XML object to parse the National Weather Service data.

In order to get the image off the Internet, Jeff modified an existing HTTP Client object and a

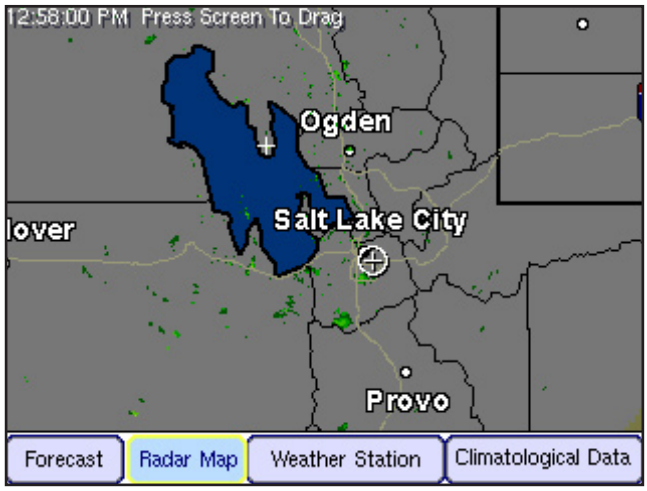


*Figure 3 - Radar Map of the Salt Lake Area*

DNS Lookup object to find the web site's IP address, and the "get" command to retrieve the image. He allocated twenty percent of the QTERM-G70's processor time to fetch the image from the Internet every fifteen minutes and perform the GIF to BMP conversion.

Since there are so many possible variations for the weather forecast (20% chance of rain, 40% chance of rain, windy, sunny, etc.), using Qlarity's standard graphic objects to display the hundreds of possible combinations could prove daunting.

Based on the data retrieved from the Internet, simple bitmap elements (sun, cloud, etc.) were dynamically assembled to represent the graphic weather forecast shown in Figure 4.

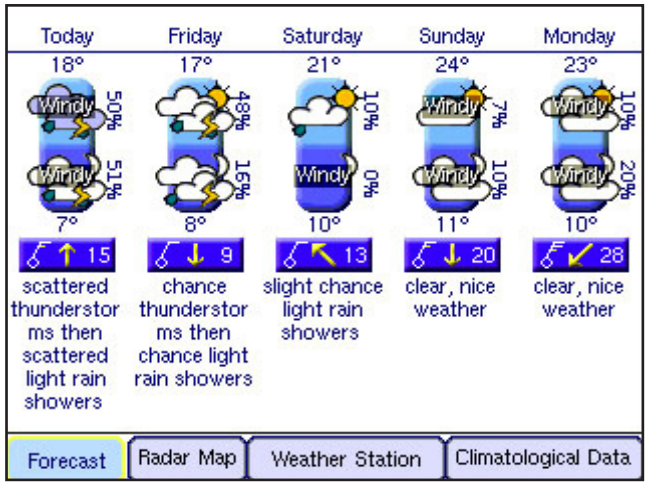


*Figure 4 - Forecast with Dynamic Graphics*

This level of information was more than adequate for all but our President. To enhance the features (i.e., "feature creep") he wanted to see historical information along with today's weather.

Jeff connected to the meteorological department of a local university and parsed a text file to access temperature, rainfall, humidity and wind speed data for the last thirty years.

Jeff parsed this data and displayed it on the screen using existing Qlarity objects such as the Chart object (Figure 5).

By using and modifying existing Qlarity objects, this robust application communicates serially to

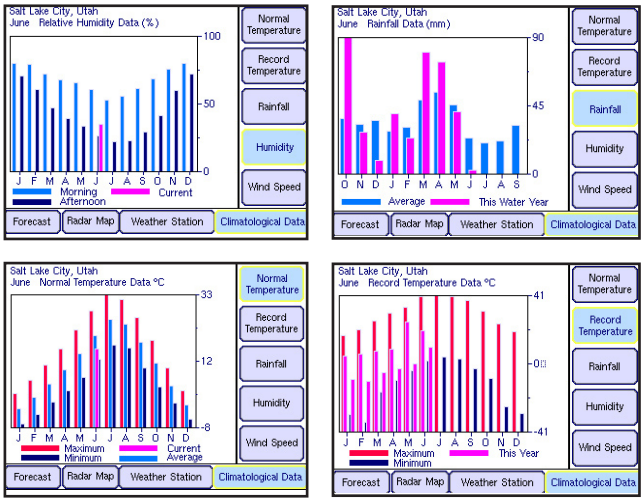


*Figure 5 - Current and Historical Temperatures*

another device and via Ethernet to our company network and accesses graphics and text data from various web sites.

To learn more about Qlarity, download a FREE copy of our GUI application builder (Qlarity Foundry) and download sample applications, visit www.qlarity.com.