Case Study

Conference Room Scheduler with Email Notification

QSI has been "eating our own dog food" (i.e., using our own terminals in real-world applications) for quite some time. We use QTERM-G70s to control the terminals being tested in our environmental chambers, as a weather station interface (see separate case study) and for a billiards room tournament ladder.

As with many companies, scheduling meetings, alerting attendees and reserving the conference room can be an arduous job. While a piece of paper with a date grid on a clipboard attached to the wall outside the conference works some of the time, we had a problem with outdated weekly schedules.

One of our software engineers has created a Qlarity application that runs on the QSI QTERM-G75 (Figure 1). This color, graphic, touch screen terminal makes scheduling meetings very easy.

Figure 1 - Main screen showing scheduled meetings

The QTERM-G75 is attached to our company network via the optional Ethernet port on the back of the terminal. The terminal queries an employee database using a *Qlarity http object*. The object reads a php page on our company intranet and retrieves the employee name, email address and department. This information is used in scheduling and alerting attendees.The employee information is stored in the terminal's flash file system and is updated once per day.

Scheduling a meeting can be done on the touch screen terminal or using a PC application we call *Virtual Qlarity*. *Virtual Qlarity* establishes a link between the PC and the QTERM-G75 and the user can see what is displayed on the terminal as well as schedule and modify meetings.

Touching a date and start time and dragging to the meeting end time creates a meeting, (Figure 2). Alternately, you can select from one of four buttons to schedule the room now ("Instant Meetings"). Meetings can be scheduled indefinitely into the future. The terminal stores meetings from the past week in RAM. The last week is also stored in the flash file system and loaded into RAM in case of power failure. Meetings for the past year are also stored in the flash file system.

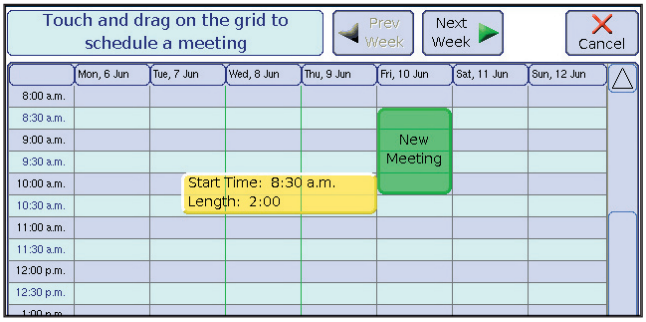

Figure 2 - Scheduling a meeting

The grid that displays the dates and times was based on[1] the existing *datagrid* object, while the meeting box and flyout were based on the existing *translucent label* object.

Once the duration of the meeting is set, the meeting is named and attendees are selected individually, by department or by a named group. The keypad shown in Figure 3 is used to filter the employee names in the list box. The grayed out letters indi-

cate that there are no employees in the database that start with that letter.

The bottom window of Figure 3 specifies when the email notifications are sent out to the attendees, the default is when the meeting is scheduled and five minutes before the meeting is to start.

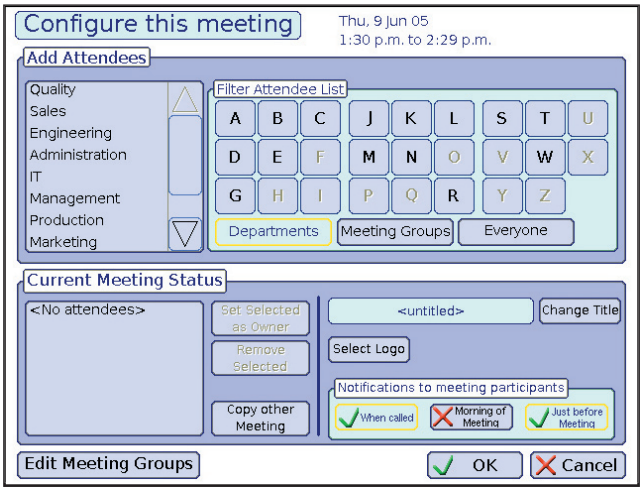

Figure 3 - Attendees selected using the touch screen.

The QTERM-G75 uses an *SMTP object* that we created to compose and send the email notifications. The terminal stores a list of meetings and initiates an email prior to the start of the meeting.

In this window are other options like copying the attendees from a prior meeting, selecting and displaying a logo for the company with whom you are meeting and creating or editing meeting groups.

Figure 4 displays a meeting in session, the start time of the meeting, the time remaining and the company logo of the company in the meeting.

Potential enhancements in the future include an audible voice that announces when the meeting is nearing the end and an interface to turn the heater and air conditioner on or off when there are no meetings in session.

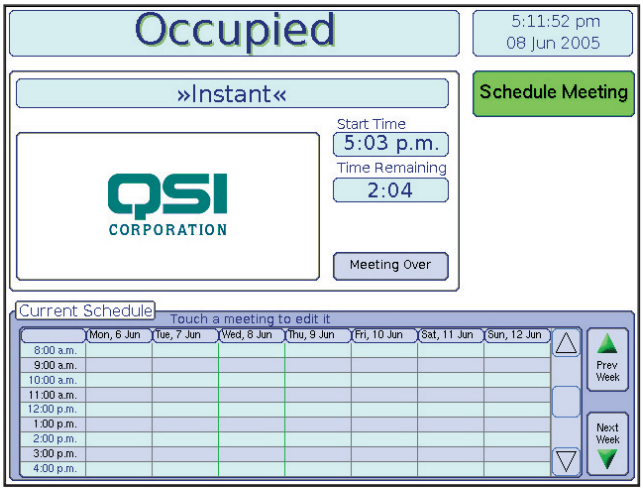

Figure 4 - Meeting in session screen

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

[1] Object Inheritance - When creating new objects that are based on other objects, there is no need to copy and maintain two code bases. Simply add the unique code to the new object and all of the base code remains in the source object. Changes in the code of the source object is inherited by the other objects.