WESTERN STATES FORUM
HAND-HELD DIAGNOSTICS TERMINAL

6/18/2009

Michael Edwards
Ryan Huffman

Overview

- Topics covered:
  - Problem statement
  - Rationale for solution
  - Software design
  - Live demonstration
  - Workflow process
  - Future developments
  - Pitfalls

So, What's the Problem??

- Currently, no quick and effective way to control ITS equipment
  - Laptop is too much of a hassle
    - Software setup is complex
    - Bulky to carry around
    - Fragile
  - Risk for Operators
    - Slow and ineffective control

What can we do?

- Find a dedicated device
- Evaluate third party devices:
  - Portable and rugged
  - Communication
    - RS-232, RS-422, RS-485, Ethernet
  - Battery Operated
  - Programmable
  - NEMA 4 Compliant

Target ITS Equipment

- CCTV (Closed Circuit Television)
  - Camera protocols: Cohu, Pelco
  - Support for
    - Pan, tilt, zoom functionality
    - Image downloading
    - Custom commands
    - Place/edit screen text

Target ITS Equipment (Cont.)

- CMS (Changeable Message Sign)
  - Support for
    - View, set, edit text for display
    - Perform diagnostics tests

Target ITS Equipment (Cont.)

- Other ITS Equipment
  - Extensibility is important!

Hand-held Terminals

QSI G-58

Two Technologies jett.XL

Possible Hardware (Hand-Held terminal)

QSI

- Pros
  - Cheaper ($600-$800)
  - Custom OS
- Cons
  - Software is not portable
  - Custom OS

Two Technologies

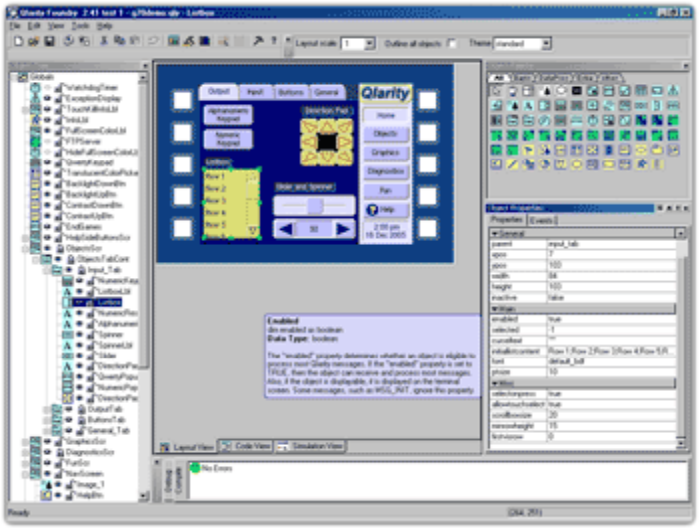
- Pros
  - Windows CE
  - Software is portable
- Cons
  - More expensive ($1000+)
  - IT doesn't like Windows

QSI G-58

- Communication interfaces
  - Software-settable EIA-232, 422, 485
  - Ethernet (TCP and UDP)
- Keypad
  - 45-button keypad
  - Back lit
- Programmable using proprietary language (Qlarity)

Qlarity Development

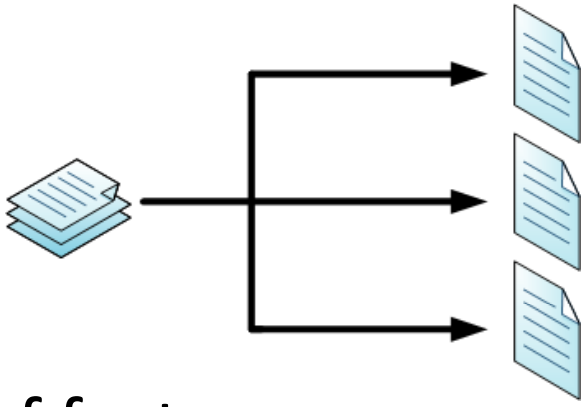
- Basic-like programming language
  - No dynamic object instantiation ☹
- Qlarity Foundry IDE
  - Code Editing
  - GUI Builder
  - Library Management
  - Software Emulation
  - Compiles and downloads to terminal



- Stand-alone compiler & downloader

Non-Collaborative IDE

- Environment stores projects in one large file
- Impossible to separate
- What did we do?
  - Use IDE for quick prototyping of features
  - Separate code outside of IDE
  - Use command-line tools to compile

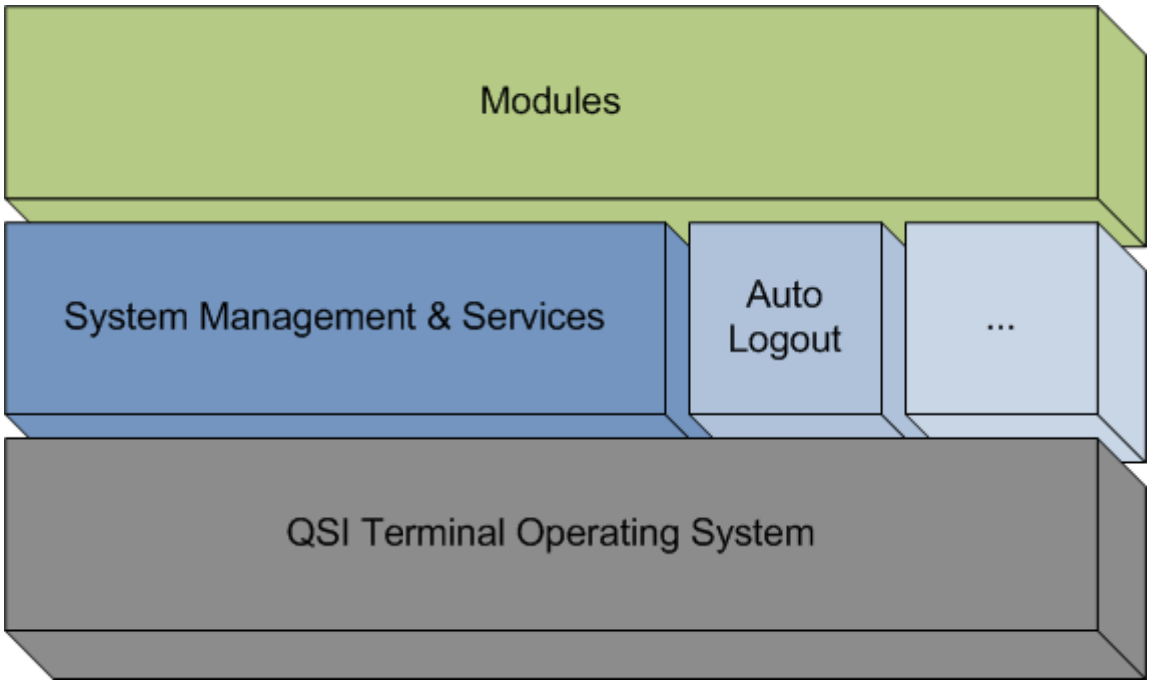


Programming Environment

- Using editors with custom highlighting
  - Vim
  - Crimson Editor
- Source code is shared and managed using Subversion

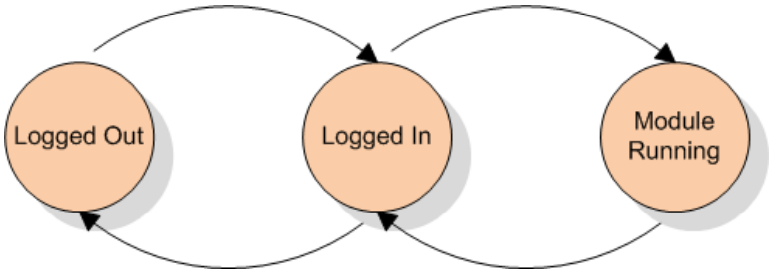
Software

- Our software solution is composed of two main parts:
  - System
  - Modules



System

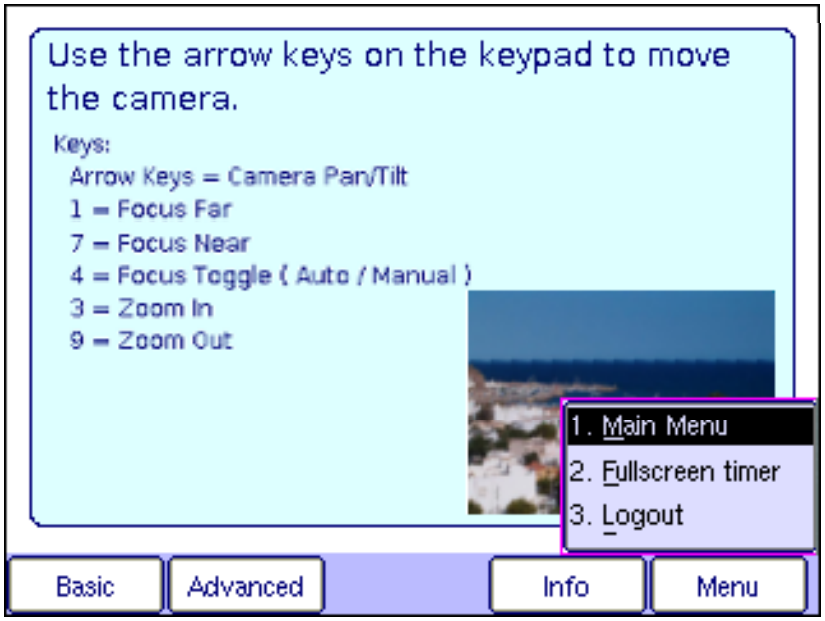
- System state manager
- Screen manager
  - Controls displaying of screens
- Other
  - Auto-Logout
  - Configuration Loader
  - Configuration Editor
  - System Updater (Future update)



Modules

- One for each piece of ITS equipment
- Makes calls to system
  - Display screen
- Developed independently
- Defines “Soft Keys”

Screen Layout



- "Soft Keys" map software actions to physical keys on the terminal. Each module defines it's own set.
- Standard GUI components (list boxes, text boxes, scroll bars, etc…)

Keypad Layout

- Top row of keys control “Soft Keys” onscreen
- Shift key toggles between upper-case (only) letters and ASCII characters.

CCTV Module

- Functionality
  - Basic PTZ
  - Message displaying
  - Custom commands
    - For manual command input.
  - Download/Display of current image from the CCTV server.

Demo

- Materials
  - Demonstration table
  - CCTV camera setup (with video encoder)
  - Bucket of water
  - QSI G-58 terminal and power and communication cables
- Demo
  - Startup Screen
  - Login Screen
  - Module & Location Selection Screen
  - CCTV Module
  - Configuration Editor

Creating a Module

- Can be created by anyone
- Requirements:
  - Learn Qlarity
  - Extend from our pre-existing templates
  - Use development guide
    - Will be provided by us in the future
  - Write code in a text editor
    - Slow and inefficient, especially for new developers
    - Is there a solution?

Use Foundry Editor!

- Foundry project files
  - Really just source code, and a lot of it
  - With tweaks, importable to system code
- Easier for new developers
  - User-friendly source code editing, including IntelliSense-like capabilities
- Easier to develop GUIs
  - WYSIWYG editor enables quick prototyping
- Tools for debugging

How We Organized Ourselves

- Being at school meant we probably didn't see each other at the same time.
  - Trac
    - List of tasks.
    - Wiki for documentation.
    - Helped us stay organized and synchronized!
  - Situated ourselves on Friday when we were both working.

Looking Ahead

- Newer models include touch screen capabilities making terminal use more intuitive
  - Non touch screen has proven to be a hassle
- Deployment of development pipeline to ease the creation of custom modules.
  - Development Manual
  - Qlarity Foundry IDE
- Expanding the basic set of modules that will be shipped with every hand-held terminal.

Pitfalls

- Non-collaborative work environment from QSI
- No effective way to decouple modules
- No dynamic object creation
  - Object naming conflicts in Qlarity
- The provided connector is weak and easily broken (12-pin Hirose)