Site-Level Specification Guide

Human-Machine Interface

This HMI Specification document provides a list of high-level requirements to assist in the specification of a distributed HMI system.

Table of contents

1 General requirements

1.1 The operator interface software, herein described as the HMI (Human Machine Interface), shall be an integrated package for developing and running automation applications. The HMI shall be designed for use in Microsoft®, Windows 7, Windows 8, Windows 8.1, and Windows 10, as well as Windows Server 2008 R2, Windows Server 2012 Standard and R2, and Windows Server 2016. It shall use COM, ODBC, OPC, and ActiveX technologies for optimal performance and integration with other software systems.

1.2 The HMI shall be based on Microsoft user-interface standards. The HMI shall:

- store all historical data in local and/or an ODBC-compliant database,
- support VBA scripting for integration with other Windows products, and
- be able to act as an OPC client to allow for data exchange with a wide range of process devices.

1.3 The HMI shall support multiple development environment clients that can have simultaneous access to the HMI application.

1.4 The HMI shall support multiple HMI servers in an application. HMI servers can also be redundant.

1.5 The HMI shall support multiple run time clients that can have simultaneous access to the HMI application.

1.6 In non-redundant scenarios, the HMI shall support up to two HMI servers hosted on a single computer.

1.7 In redundant scenarios, the HMI shall support up to one HMI server hosted on a single computer.

1.8 The HMI shall provide a common way to define and authorize secured actions on resources for a set of users or groups and locations.

1.9 The HMI shall allow for seamless integration and interoperability with other Rockwell Automation products to allow for sharing of tag data without duplication of tag databases and other functionality.

1.10 The HMI shall provide an Application Explorer for organizing and working with projects. It shall contain all editors for creating projects and shall display project files as they are created. The HMI shall include a large selection of commonly used graphic objects and library objects that can be dragged and dropped into a graphic display. The HMI shall also include a tool that enables adding symbols and addresses from a Programmable Logic Controller (PLC). All project files shall be in a directory structure that does not mix application files (user-developed project files) with system files, for easy data backup.

1.11 The HMI Application Explorer shall support editing of remote projects from different computers. This enables the separation of configuration software and run-time software which provides a more stable run-time environment.

1.12 All HMI projects should be viewable and editable from the same engineering station in the application tree.

1.13 The HMI editor should allow for simultaneous collaboration by multiple developers.

1.14 The HMI shall provide a tool to show the status of installed product patch file versions currently installed on a computer.

1.15 The HMI shall provide the ability to design high-level graphics for complex applications either by using its own drawing editor or by importing graphic files from other drawing packages such as AutoCAD®, CorelDRAW® and Photoshop™. Specifically, the HMI shall allow importing of the following file formats: WMF, .CLP, .BMP, .TIF, .GIF, .PCX, and .JPEG. The HMI shall include, but not be limited to, the following graphic object animations: position, rotation, size, visibility, color, fill, slider, and touch.

1.16 The HMI shall integrate Microsoft Visual Basic for Applications (VBA) as a built-in scripting language to customize and extend applications. The HMI must adopt Microsoft's Component Object Model (COM) and implement COM technology as a means of exposing open application interfaces to external applications, such as Microsoft Visual Basic (VB). VBA scripting will handle graphics similar to the way forms and other graphics are handled in VB. This will allow for access to all the events, methods and properties of graphical objects and ActiveX controls in the HMI.

1.17 The HMI shall provide an 'HMI Server Backup and Restore' utility that has the ability to backup running HMI servers without shutting down and restore servers into applications.

1.18 The HMI shall provide an 'HMI Distributed Application Manager' that has the ability to backup an entire distributed HMI application, without shutting down, and allows the entire application to be archived or restored elsewhere.

1.19 The HMI shall provide the ability to develop an HMI application in one location (development site) and test it before commissioning onsite, using a backup and restore tool.

2 System Sizing Guidelines

2.1 If your system needs to expand beyond these guidelines, contact your Rockwell Automation sales or distributor representative to discuss your application design.

- Alarm and Events
  - 10 non-redundant Alarm and Events servers or 10 Alarm and Events redundant server pairs per application.
  - 10,000 Controller device-based alarms per server. This is the sum of both instruction alarms and tag-based alarms across all the controllers referenced in each data server.
  - 100,000 total includes all Alarm and Events tag-based and device-based alarms per application.
- Data
  - 10 non-redundant data servers or 10 redundant data server pairs per application.
  - 1,000,000 total device tags per application.
  - 100,000 total tags per data server.
- HMI
  - 10 non-redundant HMI servers or 10 redundant HMI server pairs per application.
  - 40,000 HMI based tags per HMI server. Maximum number of HMI tags that can be created on a single HMI server is 175,000. Extensive use of HMI tags has a significant performance impact on an HMI server, so Rockwell Automation recommends minimizing their use, and instead using direct-reference tags as much as possible.
  - 5,000 total data logging of tags per HMI server (1 second scan rate)
  - 120 simultaneously connected HMI clients per application.
  - 4 ViewPoint web servers per application.
  - 50 concurrently connected ViewPoint web clients per ViewPoint web server.

3 Architecture

3.1 The graphic viewers, or HMI clients, shall be separate from the business logic, or HMI Servers, and both are separate from the configuration software.

3.2 The HMI shall support data servers to communicate with any OPC server.

3.3 The HMI clients shall be able to view tag data from any HMI server or data server in the application.

3.4 The HMI client shall be able to view displays from any HMI server in the application.

3.5 The HMI shall support direct access to control information. This eliminates the duplication of entering tag database information more than once.

3.6 The HMI shall support the retrieval and display of Extended Property values for controller tags. Extended Properties available in the controller shall include minimum and maximum value, description, engineering units, and state.

3.7 The HMI shall support the retrieval and display of Extended Property values in different languages as stored in the controller. The language to be displayed shall be selected in the HMI, but retrieved from the controller at runtime.

3.8 The HMI shall support remote editing. Any computer with sufficient security and the configuration software installed can add, change or delete any configuration information on any computer in the distributed application.

3.9 The HMI shall support a scalable design environment. The HMI shall support the migration of machine level HMI projects to site level HMI projects.

3.10 The HMI servers shall run as a service and will not have a user interface. This allows for secure headless operation and does not require a user to be logged on at the server.

3.11 The HMI shall provide support to configure and interact with the server by using the configuration software or the HMI client.

4 Security

4.1 The HMI shall use the Local Directory and/or Network Directory provided by the Security services:

- Local Directory: all project elements are located on a single computer and security information is shared with other participating software products located on the same computer.
- Network Directory: information about project elements and security is organized for multiple enabled products across multiple computers on a network.

4.2 Security shall use the following policies to define system-wide rules that govern how security is implemented:

- System Policies:
  - Security policies – define general security rules
  - Audit policies – define what security-related information is audited while the system is in use

    - User Rights Assignment policies – define which users can access particular features
    - Health Monitoring parameters – define application wide settings to tune the application and accommodate network issues for a distributed system.
- Product Policies: sets of securable features for the individual products in the system

4.3 Security shall provide a common way to define and authorize secured actions on resources for a set of users or user groups and locations.

4.4 Security Policy settings in the Network Directory shall be completely separate from those in the Local Directory.

4.5 The HMI shall provide a tool to configure Security settings.

4.6 The HMI shall provide an optional, stand-alone tool for administering a system to do the following:

- Create and configure application, area, and data server elements in a Directory.
- Backup and restore an entire directory, an individual application, or system-wide settings.
- Configure options for routing, logging, and viewing diagnostic messages.

4.7 The HMI shall have the ability to allow certain users or groups of users to access only certain parts of the system. The security shall be based on a series of codes. Each code shall allow the users or groups of users with security privileges for that code, to access the HMI commands, macros, graphic displays, OLE verb controls, and tags allowed by that code. The HMI shall allow assigning individual users combinations of security codes, allowing each user to access different sets of features.

4.8 The security system can use the Windows security system. This will closely integrate the overall system security model.

4.9 The security system shall allow the use of Windows user accounts and groups. This enables users to be added and removed from the Windows user groups without changing the HMI application.

4.10 The security system shall be able to assign each person a user account with a login name, password, and any desired macros. The desired macros execute on login and logout.

4.11 The HMI shall have a minimum of 16 different security codes.

4.12 The HMI shall have the ability to set up security by either inclusion or exclusion.

4.13 The HMI shall provide a way to evaluate a user's membership in a security group from a display screen, via faceplates, global objects, or other objects on screen.

4.14 The HMI shall provide role-based security user groups so that security privileges can be assigned to defined areas of a plant according to a user's role.

4.15 The HMI shall provide a means for operators to change their passwords while a project is running.

4.16 The HMI shall provide a confirmation pop-up dialog box from buttons, numeric inputs, string inputs, or objects with touch animation that will enable operators to confirm that they want to take the requested action, before it occurs.

4.17 The HMI shall allow the application designer to specify the location of a pop-up dialog box relative to its calling object.

4.18 The HMI shall provide increased security through electronic signature control. The signature control will require the user to enter their user-name and password, and optionally obtain verification from a supervisor, before performing a set-point change, command, or recipe download.

4.19 Optional authorization. The HMI shall not allow the update of a given set-point or the execution of a command to occur until the signature has been authorized.

4.20 The HMI shall log all signature control run-time activities to the activity log.

4.21 The HMI shall allow the Windows desktop to be locked out.

4.22 The HMI shall support auto logging out of a user after a configurable period of inactivity.

4.23 The HMI shall capture a tag value before applying a change when the change is made with a numeric or string input. Tag values both before and after the change shall be available to be logged by the system, as configured by the application designer.

4.24 The HMI shall support action groups to group different actions together and assign security permissions to all of the actions in the group.

4.25 The HMI, with Local Directory, shall allow all users to have full access to the directory by default.

4.26 The HMI, with Network Directory, shall allow all users that are members of the Windows Administrator group on any local computer that is connected to the Network Directory, to have full access to the directory.

5 Application Explorer

5.1 The HMI shall provide an Application Explorer to organize and work with HMI servers.

5.2 The Application Explorer shall be able to edit all the HMI servers in a system within the same application tree.

5.3 The Application Explorer shall support drag and drop between HMI servers in an application and between multiple copies of the Application Explorer.

5.4 The Application Explorer shall support a tree view of all the servers and their components.

5.5 The Application Explorer shall allow for editing components and testing components.

5.6 The Application Explorer shall allow for editing of all components in a running HMI system.

5.7 The Application Explorer shall support a folder hierarchy to allow the application to mimic the physical layout of the plant. Folders can be added to the second level of the application tree.

5.8 Folders can contain an HMI server, and any number of data servers.

5.9 The HMI shall provide the ability to copy HMI servers between folders without the need for renaming of components.

5.10 The configuration software shall support security that enables only valid users to view and edit data.

6 Communications

6.1 The HMI shall provide full optimization of tag writes to contiguous data held in devices, to allow quick and efficient communication on downloads to any OPC servers that provide write optimization.

6.2 The HMI shall provide communication drivers to Rockwell Automation devices at no additional cost.

6.3 The HMI shall have the ability to switch automatically to a pre-defined secondary network if the primary network fails at run time.

6.4 The HMI shall act as an OPC client. The HMI shall support both local and remote OPC connections. During configuration, the HMI client shall produce a list of all known registered OPC servers. When functioning as an OPC client, the HMI must be able to implement the OPC 'Browse Namespace' method.

6.5 The HMI shall automatically scan only required values. When a display opens, the HMI shall request information on the required points. The display will receive updates when the value changes until the display closes.

6.6 The HMI shall support directly referencing the tag in the controller. This eliminates the need to create an HMI tag and greatly reduces the amount of configuration that is required.

6.7 The HMI shall support data server redundancy. Any OPC data server can have a secondary data server associated with it. If the primary server fails, the defined secondary server will take over the OPC connection, providing uninterrupted access to data.

6.8 The HMI shall support a seamless transition during data server fail-over. The fail-over will not require any user interaction from the clients.

6.9 The HMI shall support switching back to the primary data server from the secondary, when the primary data server comes back online. Alternatively, the HMI can remain connected to the secondary data server even if the primary data server becomes available.

6.10 The HMI shall aggregate multiple data sources into a single namespace and a single connection. Clients will not need to manage individual connections to multiple data servers.

6.11 Once a data server is defined in the HMI application, it will be available to all HMI clients.

6.12 The HMI shall provide additional offline access to a data server's namespace. Access to the data server namespace will be available when the data server is offline. The data server must be online to obtain run-time values for the data items.

6.13 The HMI shall provide a way to manually inhibit data communications and alarming from devices, for maintenance or equipment change, without affecting HMI performance or stability. Communications inhibit shall be available at design time via a communication shortcut configuration dialog, and at runtime from an HMI screen with a predefined tag. The HMI shall provide a visual indication that communications have been inhibited at runtime.

6.14 The HMI shall provide a way to configure communications shortcuts to CIP Energy devices that communicate using the CIP protocol.

7 Application documentation

7.1 The HMI shall provide comprehensive documentation of an application by using a documentation utility provided with the HMI software.

7.2 The HMI shall provide the ability to scan through an entire HMI application to eliminate the need to manually create documentation.

7.3 The HMI shall provide the ability to see tag cross-references showing where both HMI tags and direct tag references with controllers are used throughout the application.

7.4 The HMI shall provide the ability to export the application documentation to an easy-to browse HTML format.

8 Tag database

8.1 The tag database shall define what HMI tags will be monitored. Each entry in the tag database shall be called an HMI tag.

8.2 The tag database shall be organized in a hierarchy, with each level represented by a folder that can be expanded or collapsed.

8.3 The HMI shall have the ability to update the current value of a tag from the device to which it is connected, and then store that value in RAM so it is immediately accessible to all parts of the HMI.

8.4 The HMI tag database shall provide four types of tags: analog, digital, string, and system. Each tag shall have the ability to receive its data via an OPC server or from memory. A tag with OPC as its data source shall receive its data through any respective OPC server. A tag with memory as its data source shall receive its data from a value table and can be used for local storage purposes.

8.5 The tag database shall provide the ability to generate tag names consisting of up to 256 characters. The tag names shall be able to contain the following characteristics: A through Z, 0 through 9, underscore (_) and dash (-).

8.6 The tag database shall provide the ability to enter a tag description, minimum value, maximum value, scale, offset, and units (if analog), on and off labels (if digital), initial value, security access code, and alarming description.

8.7 The tag database shall provide the ability to duplicate, edit, and delete any individual tag or folder of tags.

8.8 The tag database shall have the ability to selectively import tags from a PLC database. Tags imported in this way shall be copied into the database and shall not be shared with the source PLC database.

8.9 The HMI shall have the ability to modify the tag database while a project is running. That is, it shall be possible to add a tag in the run mode to the database (with alarming, data logging, and displays all active) without stopping the project.

9 Derived tags

9.1 The HMI shall have the ability to create a tag whose value is the result of an expression. The expression can be made up of mathematical operations, tag values, if-then-else logic, and other special functions. The current value of the derived tag shall be stored in an analog, digital or string tag. Multiple derived tags may reside in the same derived tag file or in up to 20 different derived tag files that run simultaneously.

9.2 The HMI shall have the ability to specify the evaluation period of the derived tag.

9.3 The HMI shall have the ability to edit derived tags during development or run time.

9.4 The HMI shall have the ability to start and stop derived tag processing while a project is running.

9.5 The HMI shall have the ability to directly write to an HMI tag or a data server tag via a derived tag.

10 Embedded variables

10.1 The HMI shall support embedded variables to display values that change dynamically at run time by putting placeholders in strings.

10.2 Embedded variables shall consist of any of the following:

- Numeric (analog or digital) tags
- String tags
- Tag placeholders
- The time
- The date

10.3 The HMI shall allow creating embedded variables in these editors:

- Graphic Displays
- Local Messages
- Information Messages
- Alarm Setup

11 Macro capabilities

11.1 The HMI shall provide a macro capability that executes system commands, user-defined commands, and other macros.

11.2 The HMI macros shall be securable. The HMI shall provide a mechanism to restrict certain users from executing given macros.

11.3 The HMI macro capability shall permit parameter passing of up to seven variables.

11.4 The HMI macro capability shall permit macros to call macros.

11.5 The HMI macro capability shall permit synchronous or asynchronous operation.

11.6 The HMI macro editor shall be a simple text editor permitting other editors to create macro files when necessary.

12 Alarms and Events

12.1 The HMI shall allow users to set up a complete Alarm and Events system

12.2 The Alarm and Events system shall support configuration of sounds and status tags for alarm priorities

12.3 The Alarm and Events summary display can contain tags from all servers in the application, and can be viewed from any client in the application.

12.4 The Alarm and Events system shall provide a way to hide or show out-of-scope alarm occurrences at runtime. This function shall be available to the operator via an alarm display object, and shall be available as a function of VBA for the application designer. Alarm and Events shall support computer tag-based alarming and controller device-based alarming.

12.5 Alarm and Events summary objects shall be able to be embedded on any display. It shall be possible to include or exclude any tag in a summary by the use of intelligent filters, which must be of the following types:

- wildcard (* or ?) constructs
- tag types (analog, digital)
- alarm states ('Faults', 'Out of Alarm' and 'Only show current tags in alarm')
- severity (levels 1-8).

12.6 The Alarm and Event system shall allow the operator to write a custom message to the alarm history.

12.7 The Alarm and Event system shall allow the operator to call a command or macro when an alarm in the summary is selected. This functionality must pass the following alarm information as comma- separated parameters:

- tag name
- alarm type

- severity
- value
- date
- time
- tag type

12.8 The Alarm and Events system shall have a minimum of eight alarm thresholds capable of dynamically changing during run time via tags. The alarm system shall be able to generate alarms on an increasing threshold, decreasing threshold, or both. It must be possible to disable alarm generation when approaching normal operating range.

12.9 The Alarm and Events system shall have the ability to use variable thresholds that are changeable at run time.

12.10 The HMI shall support up to ten Alarms and Events Servers for computer tag-based alarms and for controller device-based alarms.

12.11 Tag-based alarming shall provide monitoring for a system that includes controllers, which communicate through OPC-DA servers or if not using the pre-built alarm instructions in controllers.

12.12 Device-based alarm monitoring shall utilize pre-built alarm instructions, available in some controllers. The controller detects and monitors alarm conditions, keeping all alarms and event processing in the controller.

12.13 The two Device-based alarm instructions in some controllers are:

- ALMD – Boolean alarms
- ALMA – Analog alarms

12.14 The pre-built instructions shall be programmed in the controller only once, reducing programming effort and errors.

12.15 Device-based alarm monitoring shall eliminate the need for duplicating alarm tags in an HMI server and controller.

12.16 Device-based alarm monitor shall reduce controller communication resources and network overhead by eliminating alarm polling. The alarm status is communicated only when the state changes.

12.17 Device-based alarm monitoring shall provide accurate time stamps on alarm conditions that are generated from controllers.

12.18 Device-based alarm state shall be managed, processed, and preserved by controllers, even if a computer goes down.

12.19 Alarms and Events shall support language-switching with alarm messages.

12.20 Alarms and Events shall support secure access to alarm and event operations through integration with Security.

12.21 Alarms and Events shall support up to 20,000 computer tag-based alarms and up to 10,000 controller device-based alarms, to a total of 100,000 per application.

12.22 The Alarms and Events shall support associating up to four tags with each alarm to include process data with event information and alarm messages.

12.23 The Alarms and Events shall support associating a classification string or alarm class to alarms. The alarms shall support filtering with this alarm class string in the alarm viewer objects.

12.24 The Alarms shall support associating a command, up to 1,000 characters long, with the alarm.

12.25 The HMI shall support a logging component that manages connections between alarm servers and databases and logs data from each alarm server to an alarm history database.

12.26 The Alarm and Event logging shall be used with a Microsoft SQL database.

12.27 The Alarms and Events shall support objects to view and analyze alarms during runtime.

12.28 An Alarm and Event Log Viewer shall allow viewing, filtering and printing data from alarm history databases. Other database tools can also retrieve, view, analyze, and print alarm history information.

12.29 The Alarm and Event Log Viewer shall allow custom color coding of alarms displayed, based on alarm state and priority.

12.30 The Alarms and Events shall support custom filters on the Alarm View object that shall be editable at runtime.

12.31 The Alarm and Event Summary object shall be highly configurable. It shall support operator interaction to acknowledge, disable, suppress, filter, and sort alarms during run time.

12.32 The Alarm and Event Banner object shall be used to monitor and respond to the most serious alarms requiring immediate attention.

12.33 The Alarm and Event Banner shall display only the highest priority, most severe and most recent alarms.

12.34 The Alarm and Event Banner shall allow the application designer to choose how alarms are sorted in the banner at runtime. Alarms can be sorted by either Alarm State-Priority-Event Time or Event Time-Priority.

12.35 The Alarm Status Explorer object shall be used to enable or disable alarms, suppress or unsuppress alarms, and view operator comments.

12.36 The Alarms and Events objects shall subscribe to alarms and events from one or more areas in the system.

12.37 The Alarms and Events tag-based alarms shall allow organization of the alarms into logical groups of up to 4 hierarchical layers.

12.38 The Alarm Summary, Alarm Banner, Status Explorer, and Alarm Log Viewer shall allow tag-based alarms to be viewed, filtered, and sorted by alarm group name.

12.39 The Alarm and Event system shall provide function and expression commands that can be used to display information about current alarms, including: Disabled Count, In Alarm Shelved Count, In Alarm Suppressed Count, Shelved Count, and Suppressed Count. These allow the application designer to display information about the Alarms and Events system on displays throughout the application.

12.40 The Alarm and Event system shall implement a handshake mechanism between the HMI and the PLC ladder logic that guarantees latched alarms are recorded in the alarm history file. The alarm system must be flexible in this implementation to allow the PLC ladder logic to reset the bit, or to manage the resetting of the bit, in the HMI.

12.41 The HMI will allow the user to define which computer in the application contains the central log. Log information from all computers in the application will be logged to the central log.

12.42 The HMI will allow the periodic central logging interval to be defined by the user, in terms of seconds, minutes, hours or days.

12.43 The HMI will have the ability to create a table in the database for use as central alarm log storage.

12.44 The HMI will have the ability to define a user name and password for the central log database connection.

13 Data logging

13.1 The HMI shall have the ability to record specific tag values under certain conditions. Several models shall define these conditions. The collected data shall be stored in a file or ODBC database for displaying in trends, for archiving for later processing or analysis.

13.2 The HMI shall have the ability to start and stop data logging while a project is running.

13.3 The HMI shall be able to run 20 different datalog models simultaneously.

13.4 Datalog models must support three different data collection modes: periodic polling, on change, and on demand. The periodic polling rate must have configurable time units of hundredths, tenths, seconds, minutes, hours or days. The default time unit will be seconds.

13.5 The datalog system shall have the ability to create data log files periodically, at specified times, on event, and never. The datalog system shall have the ability to automatically purge old files or records after a specified time.

13.6 The HMI shall be able to switch to an alternate path if a problem is encountered logging to the primary path. Errors such as 'low disk space' and 'dropped network connections' must be handled automatically.

13.7 The HMI shall provide a way to automatically merge data logged to an alternate location back to the primary location when the problem is corrected.

13.8 The HMI shall provide a mechanism for renaming the historical data set. It must be possible to link the historical data with a batch or lot ID.

14 Activity logging

14.1 The HMI shall have the ability to record information about various types of system activity. This information shall be stored in the Windows Event file or an ODBC database for archiving for later processing or analysis, and/or for displaying and analyzing with other software tools, such as Dream Reports and Microsoft Excel.

14.2 The HMI shall have the ability to log message received from a machine level HMI to have a central log of all HMI activity.

14.3 The activity log shall have the ability to log any of the following: command and macro usage, operator comments, system messages and errors, communication network errors, tag read and write activity, and custom messages.

14.4 The activity log shall allow designating when to clear or overwrite entries and what activities to llog.

14.5 The HMI shall have the ability to edit activity logging during development or run time.

14.6 The HMI shall have the ability to conditionally log information to a hard disk and to a status bar by category.

14.7 The HMI shall present the activity information in a dockable window that can be dynamically resized and scrolled. This activity bar can be re-docked or disabled during run time.

14.8 The HMI shall have the ability to view the activity log from any computer in the application. The HMI shall support periodic central logging of activity log file information to an ODBC-compliant database. Central logging will support, but not be limited to, Microsoft Access, Oracle, and Microsoft SQL Server.

14.9 The HMI will allow the user to define which computer in the application contains the central log. Log information from all computers in the application will be logged to the central log.

14.10 The HMI will allow the periodic central logging interval to be defined by the user, in terms of seconds, minutes, hours or days.

14.11 The HMI will have the ability to create a table in the ODBC database for use as central activity log storage.

14.12 The HMI will have the ability to define a user name and password for the central log database connection.

14.13 The HMI will have the ability to log the old value, new value and comment for a tag write.

14.14 The HMI will have the ability to add a comment for a command.

15 Local messages

15.1 The HMI shall support local messages to provide ongoing information on the status of devices and processes.

15.2 The HMI shall provide a Local Messages editor to create local messages and specify their trigger values.

15.3 The HMI shall display the message with the trigger value that matches the connection value assigned.

15.4 The local message display shall show one message at a time from the message file assigned to the local message display.

15.5 The HMI shall support up to 10,000 messages in each message file.

16 Events

16.1 The HMI shall have the ability to trigger actions based on an event going from false to true using an expression. An expression is an equation that contains tag values, mathematical operations, if-then-else logic, or other functions. An action shall have the ability to produce a variety of functions including, but not limited to, initiating a snapshot of tag values, displaying an error screen, and changing a tag value.

16.2 The HMI shall have the ability to specify the evaluation period of events.

16.3 The HMI shall have the ability to edit events during development or run time.

16.4 The HMI shall have the ability to start and stop event processing while a project is running.

16.5 The HMI shall have the ability to run 20 event files simultaneously.

17 Graphic displays

17.1 The HMI shall provide a number of ready-made graphic displays as libraries.

17.2 The HMI shall provide a graphic display editor for creating displays using graphic objects.

17.3 The HMI shall provide, at minimum, the following toolbars:

- graphics
- objects
- alignment
- states
- pattern style
- foreground color
- background color

17.4 The graphic display editor shall have the ability to drag and drop objects from a pre-configured graphics library, paste objects that are copied to the clipboard from another Windows application such as Microsoft Word and Microsoft Paint, and insert objects created by another Windows application using OLE. True OLE support is required in that it shall be possible to call up the native application that created the object being inserted and use the native object editing tools from within the HMI.

17.5 The HMI shall provide, at minimum, the following productivity tools: easy access to the object Property Panel, Object Explorer, grid settings, ability to import other graphic formats, zooming, and drag-and-drop features.

17.6 The HMI shall allow creating a new graphic display, adding existing displays from another HMI application, generating complete graphic displays from an XML file, and exporting complete graphic displays to an XML file for modifications done using other text editor tools. The HMI shall allow single-display or multiple-displays-batch import.

17.7 The graphic display shall support VBA scripting similar to forms in Visual Basic. The HMI shall allow VBA to access 3rd party ActiveX controls and their accompanying properties, methods and events.

17.8 The graphic display native objects shall support properties, methods and events that are accessible via VBA scripting.

17.9 The graphic display shall provide support for manipulating HMI Tag database folders and tags using VBA scripting.

17.10 VBA script shall be embedded in the applicable graphic display file, and can be included in export of the graphic file to XML. This allows the application designer to export VBA along with the display, edit it in XML, and import it back into the project.

17.11 The objects shall support tooltips that can be displayed at runtime. The tooltips shall support embedded variables and be language translatable.

17.12 The HMI development environment shall allow the HMI designer to control what objects or object groups are visible while editing a display. This allows easier maintenance of complex displays that contain overlapping or stacked objects and groups. This design-time visibility shall not affect object visibility at runtime.

17.13 The HMI shall allow a tag or expression to control whether the state of a button is enabled or disabled. The HMI shall allow the runtime appearance of the button to be configured to reflect its enabled or disabled state.

17.14 The HMI shall allow the application designer to determine whether a rectangular area should be used as the touch animation surface area, or whether an irregular, graphics-based area should be used. This results in better control of touch animation surface area for the operator, and allows the designer to avoid overlapping touch-based selection areas.

17.15 Toolbars and color palettes shall be docked in optional positions.

17.16 It shall be possible to customize the color palette. Graphics drawn with a customized color palette shall not require the customized color palette to be present on all run-time computers. Colors must be stored internal to the graphic files as Red, Green, Blue numbers, not as palette indexes.

17.17 The graphic display editor shall have context-sensitive "right-click" support on all objects.

17.18 The graphic display editor shall have, at minimum, the following drawing tools and objects: snap, grid, arc, ellipse, freehand line, line, polygon, polyline, rectangle, rounded rectangle, wedge, and text.

17.19 The graphic display editor shall have, at minimum, the following editing tools: undo, redo, cut, copy, paste, delete, duplicate, tag substitution, flip, rotate, resize, reshape, align, group, ungroup, send to back, bring to front, fill, and color.

17.20 The graphic display editor shall have as a minimum the following viewing tools: zoom in, zoom out, pan, and view entire graphic.

17.21 The graphic display editor shall have the ability to use tag placeholders to provide a way to use one graphic display to represent a number of similar operations.

17.22 The graphic display editor shall provide, at a minimum, the following advanced objects:

- Push buttons, macro buttons, ramp buttons
- Numeric display and numeric entry objects
- Control List Selector and Piloted Control List Selector objects
- Numeric and string pop-up scratchpads and keypads
- String display and entry enable objects
- Local message display
- Alarm, diagnostics log, and information message objects
- Time and date objects
- Image object
- Display navigation objects
- Print display button
- List, symbol, and multi-state indicators
- Gauges, bar graphs, scales and trends
- Alarm banner, alarm list and alarm status list objects
- Recipe objects
- ActiveX control objects

17.23 The HMI shall provide the option to print displays to a PDF file, so that the operator can save screens to be referenced later or share them with others.

17.24 Numeric and String keypads provided by the HMI shall allow configuration of information shown on the keypad at runtime, including: current tag name and value, entry min/max (when configured), tag description, and tag units.

17.25 The appearance of Numeric and String keypads provided by the HMI shall be configurable, so that they can match the color and style of the general application.

17.26 Numeric and String keypads shall have the ability to display a custom caption, as defined by the HMI application designer.

17.27 Numeric and String keypads shall be able to use Extended Properties as stored in the controller at runtime, including tag Description, Min and Max.

17.28 The graphic display editor shall have the ability to create a screen background by converting objects to wallpaper. These wallpaper objects cannot be selected or edited.

17.29 The graphic display editor shall allow the embedding of hyperlinks on a display, so that the HMI application designer can include a link to external web sites or documents hosted on web servers in the runtime application.

17.30 The graphic display editor shall allow creating libraries of graphic objects.

17.31 The graphic display editor shall provide a library of graphical images that can be copied on to a display screen and animated.

17.32 The graphic display editor shall allow assigning animations to any object or group of objects. It shall also allow drilling down in a group to modify any object or object attribute without losing any object animations.

17.33 The graphic display editor shall allow animations to be copied from any object to another object.

17.34 The graphic display editor shall provide a text search and replace capability on an object or group of objects. This capability shall allow whole tags or parts of tags to be replaced on an individual confirmation or replace-all basis.

17.35 The graphic display editor shall permit 1,000 editing operations to be undone and redone.

17.36 The graphic display shall permit specifying display placement anywhere on the screen.

17.37 The graphic display shall permit specifying display size.

17.38 The graphic display shall provide at least two different display types:

- Replace, which automatically closes all opened windows
- On Top, which allows the graphic display to display in front of any other display of On Top or Replace types. The On Top display shall also be possible to be kept opened and on top when a Replace display opens at run time.

17.39 The graphic display shall provide an onscreen keyboard for data entry on systems that do not have attached keyboards.

17.40 The graphic display shall provide input fields that continuously update, showing the results of downloads.

17.41 The graphic display editor shall provide the option to load a screen into memory but not display it to the operator. This feature allows embedded ActiveX controls to process logic without being seen.

17.42 The graphic display shall allow a configurable title and system menu bar.

17.43 The graphic display shall allow selecting a color for the following items:

- Background
- Input background/text
- Interactive objects highlight
- Object with input focus

17.44 The graphic display shall provide for the inclusion of security, inherited from the HMI security system.

17.45 The HMI shall provide an editor to create multiple local message files each containing up to 10,000 messages and associated trigger values.

17.46 The HMI shall allow docking displays to an edge of the run time client window, during runtime. The area of the run time client window available for other displays shall be reduced by the height and width of the docked display. The display shall support docking on the top, bottom, left or right edges of the run time client.

17.47 The HMI shall provide the option to display numeric values with digital grouping, i.e. 5000000 vs. 5,000,000 in order to clarify values for operators. The digital grouping character shall use the numeric format of the current runtime language.

17.48 The HMI shall provide the option for the application developer to define the number of decimal places to display for numeric values. The number of decimal places can be defined by a numeric value, tag, or parameter.

17.49 The HMI shall provide the option to format numeric values in scientific notation.

18 Control of graphic displays

18.1 The graphic display editor shall have the ability to attach, as a minimum, the following animation to objects: blinking colors, visibility, rotation, horizontal and vertical movement, resizing width and height, fill, and touch.

18.2 The graphic display editor shall allow dragging of the object to visually set the range of motion for an object.

18.3 The graphic display editor shall have the ability to attach OLE verb control to an OLE object.

18.4 The graphic display editor shall have the ability to attach control that links an object or display to a key or mouse button, so operators can perform an action by pressing a key or mouse button.

18.5 The graphic display editor shall have the ability to simulate run time with test display capabilities without changing to run time.

18.6 The graphic display editor shall enable automatic scaling of displays to make the porting of applications easier between systems with different display resolutions.

18.7 The graphic display editor shall provide functionality to execute a macro or script on display shutdown, assigned on an individual display basis.

18.8 The graphic display editor shall provide a 'Command Wizard' to help users construct HMI commands when attaching animation to graphic objects.

18.9 The graphic display editor shall provide the ability to print any graphic whether running or not. If the graphic is not running, the command will wait for the graphic to collect all run-time data and then send the results to the printer. It must be possible to print a graphic in the background, never disturbing the view of the currently running screen.

18.10 A graphic display, when hosting an ActiveX control, shall allow access the properties, methods and events of the ActiveX control through VBA scripting.

18.11 The graphic display editor shall provide real-time trends that always include a minimum of 15 minutes of data.

18.12 Each graphic shall have a configurable update rate to specify the maximum rate at which data servers will send data to the tags used in the display.

18.13 The graphic display editor shall offer a quick and convenient way to view the hierarchy of objects in a display to show a tree view of all objects in the display, hierarchy of objects within groups, and highlighting objects by selecting an object type, animation type or tag name.

18.14 The graphic display editor shall have the ability to display values that change dynamically at run time by inserting embedded variables into text captions on graphic objects and in message text.

18.15 The HMI shall have the ability to use the same graphic display with different sets of tags by assigning tag placeholders to objects instead of tag names and assigning a parameter file to the graphic display.

18.16 The parameter file shall define the tags that the graphic display uses at run time.

18.17 The HMI shall provide the ability to test run a display using parameter files or lists in the design environment, allowing for screen validation without needing to test in runtime.

18.18 The HMI shall provide the ability to generate complete graphic displays from an XML file, or to export complete graphic displays to an XML file.

18.19 The XML import operation shall allow you to:

- create a new graphic display
- modify the display settings of one or more graphic displays
- create new graphic objects on a new or existing display
- update existing graphic objects on a display
- modify the following aspects of graphic objects, whether new or updated:
- properties
- connections
- animations
- groupings
- wallpaper attribute
- key assignments
- import multiple XML files as a batch in a single step.

18.20 The XML export operation shall allow you to:

- create an XML file that completely describes a graphic display
- export multiple XML files in a single step.

19 Trends

19.1 The HMI shall have real-time and historical trending capabilities. It shall also have the ability to display both real-time and historical data at the same time on the same trend.

19.2 The trending feature shall allow for basic statistical and sampling calculations on trend data.

19.3 The trending feature shall be able to plot data for as many as 100 tags on a single chart, with the ability to make pens visible or invisible during development and run time.

19.4 The trending feature shall have an object model that is accessible via the graphics VBA scripting to access the trend object at run time.

19.5 The trending feature shall have the ability to add or remove tags at run time, either individually or in groups.

19.6 The HMI shall provide drag and drop and ad-hoc trending features at runtime.

19.7 The trending feature shall have the ability to view data points from multiple historical datalog models at the same time.

19.8 The trending feature shall allow for the export of trend data to a CSV file at runtime, allowing the operator to easily save data for later analysis without requiring an external tool.

19.9 The trending feature shall have the ability to perform X-Y plots as well as time plots.

19.10 The trending feature shall have the ability to automatically best fit the Y-axis data, as well as use the minimum and maximum values of a tag. It shall also have the ability to control the Y-axis with a tag value.

19.11 The trending feature shall have the ability to scale all pens on the same Y-axis, unique Y-axis per pen, and isolated pens on the same trend.

19.12 The trending feature shall allow for a logarithmic scale.

19.13 The trending feature shall display the pen's date, time, and value as a tool tip when the mouse is held over a pen.

19.14 The trending feature shall have the ability to load predefined profiles and the ability to switch profiles during run time.

19.15 The trending feature shall display time ranges with millisecond granularity and have the ability to use either standard time or military time.

19.16 The trending feature shall have the ability to pause, resume, and automatically pause when scrolling back in time.

19.17 The trending feature shall support the following refresh rates: milliseconds, seconds, minutes, hours, and on change.

19.18 The trending feature shall have the ability to display a legend. The legend shall be able to display the full tag name, short tag name, or tag description, and be changeable at run time.

19.19 The trending feature shall have the ability to display data logged to either a primary or alternate server.

19.20 The trending feature shall support panning and click-and-drag zooming through data.

19.21 The trending feature shall support printing of only the trend and legend and not the entire display.

19.22 The trending feature shall contain a configurable real-time data buffer.

19.23 The trending feature shall support parameter passing to simplify trend and display management.

19.24 The HMI trend shall support Historian data.

19.25 The HMI trend shall support Alarm and Event messages and Diagnostics messages.

20 Expressions

20.1 The HMI shall have the ability to compare data to other values, combine data with other data, and create cause-effect relationships with other data.

20.2 Expressions shall have the ability to be used, at a minimum, in any one of the following:

- Graphic displays
- Alarm setup
- Information setup
- Macros
- Global connections

20.3 Expressions shall have the ability to be built from, at a minimum:

- tag values
- constants
- mathematical, relational, logical and bit-wise operators
- built-in functions
- if-then-else logic.

20.4 Expressions shall have the ability to be used, at a minimum, in any one of the following: graphic display, derived tag, event, activity log, data log, or any alarm.

20.5 The expression editor shall have the ability to use efficiency tools like cut, copy, and paste to produce like expressions.

20.6 The expression editor shall have the ability to use, at a minimum, the following arithmetic operators: addition, subtraction, multiplication, division, modulus, and exponent.

20.7 The expression editor shall have the ability to use, at a minimum, the following relational operators: equal, not equal, less than, greater than, less than or equal to, and greater than or equal to.

20.8 The expression editor shall have the ability to use, at a minimum, the following logical operators: AND, OR, and negation.

20.9 The expression editor shall have the ability to use, at a minimum, the following bit-wise operators: AND, inclusive OR, exclusive OR, right shift, left shift, and complement.

20.10 A command wizard shall facilitate creating actions that trigger when the expression evaluates true.

21 Process faceplates

21.1 The HMI shall provide process faceplates to help reduce development time. These displays along with corresponding images shall be easily added to the HMI application and quickly connected to instructions to enable setting up and running with minimal effort.

21.2 The HMI shall provide sets of faceplates that work with instructions, a consistent look and feel across all displays, and global objects that can be re-used in regular displays.

21.3 The built-in process faceplates shall be translated to English, French, Spanish, German, Chinese and Japanese.

21.4 Process faceplates shall work with the following instructions:

- Enhanced PID (PIDE)
- Discrete 2-State Device (D2SD)
- Discrete 3-State Device (D3SD)
- Totalizer (TOT)
- Enhanced Select (ESEL)
- Alarm (ALM)
- Alarm Analog (ALMA)
- Alarm Digital (ALMD)
- Ramp/Soak (RMPS)
- PhaseManager(PhaseManager)
- Internal Model Control (IMC)
- Coordinated Control (CC)
- Modular Multivariable Control (MMC)

22 Recipe management

22.1 The HMI shall have the ability to perform recipe management functions. The recipe system shall allow uploading and downloading recipe files.

22.2 The HMI shall provide a recipe manager that will allow the HMI designer (at design time) or operator (at runtime) to create, perform basic edits, and preview recipe values. The recipe manager shall also provide the ability to import and export recipes in CSV format.

22.3 The recipe system shall allow issuing a command before a recipe is downloaded and after downloading has completed.

22.4 The recipe system shall allow verifying communications before downloading, verifying the download completed successfully, and logging errors if a download fails.

22.5 The recipe system shall allow previewing a download before it occurs and adjusting the values before downloading.

22.6 The HMI shall provide commands that allow the programmatic upload or download of a recipe, or the editing of a recipe via the recipe manager, at runtime.

23 Language switching

23.1 The HMI shall provide the ability to configure multiple language versions of an application and switch application languages dynamically at run time, with a specified language at startup, which could be changed while the application is running on the client using the Language command.

23.2 The HMI shall also allow the application to be opened in a particular language for development purposes.

23.3 The HMI shall support a project default language.

23.4 The HMI shall support up to 40 languages in each HMI application.

23.5 The HMI shall have the ability to add or remove languages when developing HMI applications.

23.6 The HMI shall provide the option to support multiple languages in the library displays.

23.7 The HMI shall allow language switching of the application in the design environment.

23.8 The HMI shall allow for language-specific display text to be edited in design time, with text changes persisted in the current language locale. This will allow for editing of text in multiple languages for a display, with no need to export and edit strings or open and close the application in the design environment.

23.9 The HMI shall allow the selection of desired languages to be included in the run-time application.

23.10 The HMI shall allow exporting user-defined text strings into an editable file in UNICODE format and later importing it back with translated strings. The HMI shall be able to export and import user-defined string such as the title bar, labels and captions in graphics displays, and embedded string variables.

23.11 The HMI shall provide a string spreadsheet editing feature to export and import language switchable strings to and from Excel for editing or translation. The string spreadsheet editor shall support the Optimized Strings feature, which enables one-time editing to recurring strings.

23.12 The HMI shall provide a language function that returns the ID of the current run-time language.

24 Global objects

24.1 The HMI shall allow linking the appearance and behavior of a graphic object to multiple copies of that object in the same application to help develop and maintain repetitive objects more easily. When changes are made to the base objects, the copies shall be changed accordingly.

24.2 The HMI shall provide a global object display editor for creating global objects.

24.3 The HMI shall allow creating a new global object display, adding existing displays from another HMI application as global object displays, and importing XML information exported from another global object display or a standard display into an existing global object display.

24.4 The HMI shall have the ability to create base objects from all standard objects except for ActiveX controls.

24.5 The HMI shall provide the ability to drag and drop global objects onto standard displays to create reference objects.

24.6 When changes are made to the base object, its reference objects shall inherit the same changes.

24.7 Reference objects shall display the following properties in the Property Panel:

- Common properties: name, size, position, and visibility
- A state property if applicable, and
- Link properties: animation, connections, and size.

24.8 The HMI shall be able to modify the link properties of a reference object to determine which information the reference object will receive from the base object.

24.9 The HMI shall be able to break the links between the reference objects and its base object. This object shall become a standard graphic object.

24.10 The HMI shall provide default values for the link properties assigned to a reference object when it is created. These default values shall be editable as well.

24.11 The HMI shall allow parameters to be defined for global objects. This allows the use of global objects without having to break the link to the base object's tags and expressions. Structured tag support shall be available.

24.12 The HMI shall provide faceplates that support instructions available as global objects.

25 Interoperability

25.1 The HMI shall be based on standards that allow the HMI's data to be accessed, shared among, and be fully interoperable with other Windows applications.

25.2 The HMI's graphic displays shall be containers for ActiveX controls and be able to access the controls' methods, properties and events via VBA scripting.

25.3 The HMI shall log all data in files in an ODBC database for easy retrieval in other programs such as Microsoft Excel.

25.4 The HMI's database editor shall allow browsing the ladder/control scheme documentation for direct tag import. This capability shall consist of a wizard that enables on-the-fly tag creation without recompiling.

26 Networks

26.1 The HMI shall use Windows security to support a centralized and secure user name and password repository.

26.2 The HMI shall support a Windows 2008 and higher domain.

26.3 The HMI shall support a Windows Workgroup.

27 Client/server operation

27.1 The HMI shall permit client/server operation whereby graphics functionality (full operations) shall be provided at a client station.

27.2 The HMI server shall leverage Microsoft Windows 2008 and higher technology.

27.3 An application shall support multiple HMI servers.

27.4 An application shall support multiple client connections at any one time.

27.5 An application shall support multiple Data servers.

27.6 HMI client/server operation shall use Microsoft ActiveX technology and standard Internet Information Services (IIS) functionality.

27.7 The client shall support Windows 7 Professional and higher technology.

27.8 The HMI clients/servers shall implement security via Microsoft's Windows security model, hence a Windows Primary Domain Controller may be required to facilitate this.

27.9 Clients can run on a computer that is remote of the HMI server computer. Performance shall not degrade in terms of display refresh or display call-up. (For example, displays on the client refresh at a rate equal to or better than the rate possible on the server).

27.10 The client software shall use a set-up wizard for initial configuration. The set-up wizard will allow the user to decide whether the client has full access or view-only operation.

27.11 Two forms of client licensing schemes will be available: Dedicated Clients (require a license file on the client) and Floating Clients (require a license file on the HMI server computer).

27.12 The client computers must cache copies of the HMI project graphics locally and be able to automatically copy the latest graphic from the server(s) as required.

27.13 The client shall support auto downloading of an ActiveX control if the ActiveX control is not present on the client or the version has changed.

27.14 The HMI Client shall support the use of Terminal Services to lower the total cost of ownership.

27.15 The HMI shall support run-time edits, changes that take effect immediately and changes that require a non-disruptive action, such as reopening a graphic display. Some of the run-time edits include tag edits and alarm edits.

28 Redundancy

28.1 The HMI shall support redundant HMI servers.

28.2 The HMI shall support redundant Data servers, both native and other OPC Data Servers.

28.3 The HMI shall support redundant Alarms and Events servers.

28.4 The HMI Development Environment shall be used to set up redundancy options for HMI and Data servers.

28.5 The HMI shall support seamless fail over from a primary server to a secondary server.

28.6 The HMI shall support seamless fail back from a secondary server to a primary server.

28.7 The HMI shall support switching back to the primary data server from the secondary, when the primary HMI server comes back online. Alternatively, the HMI can remain connected to the secondary data server even if the primary HMI server becomes available.

28.8 The HMI shall support notification of a service disruption including computer name of failed server.

28.9 The HMI shall support notification service recovery including the computer name of active server.

28.10 The HMI shall support the replication of runtime changes made in the primary HMI Server to the secondary HMI Server from the HMI development environment.

28.11 The HMI shall support a Server Status dialog box to check the current state of a server, change a server's redundancy options, or manually switch between an Active sever and a Standby server.

28.12 The HMI shall support a controlled, manual switchover from the Active server to the Standby server

28.13 The HMI shall support the use of HMI Development Environment for manual starting and stopping of Startup components on both Primary and Secondary HMI servers.

28.14 The HMI shall ensure the server is not put into an active state until all subsystems are ready to provide data.

28.15 The HMI shall provide policy system settings to accommodate for network issues, such as the time to recognize a network glitch and how frequently a check for network failure is detected.

28.16 The HMI shall provide VBA Display Client Object Model methods to determine the state of the Primary and Secondary servers.

28.17 The HMI shall support redundancy without the need to write application logic.

28.18 The HMI shall support the ability to run an event on failover.

28.19 The HMI shall support the ability to run an event on failback. The HMI shall automatically synchronize the alarm state information on the primary and secondary so there is no disruption or loss of alarm state information on a failover.

29 Activations

29.1 The HMI shall also support a secure, software-based system for activating software products and managing software activation files.

29.2 The HMI shall support activation via phone, email or website.

29.3 The HMI shall provide a 7-day grace period activation.

30 Remote Monitoring with Web-based HMI clients

30.1 The HMI shall support fully scalable and animated web displays of existing HMI applications from the office, home, or on the road via an internet browser.

30.2 The HMI web displays shall support HTML5 browsers, including Internet Explorer, Chrome, and Safari.

30.3 The HMI application designer shall have the ability to publish HMI displays in an application that are for use only with a web browser; these displays may or may not contain content identical to displays in a full HMI client application.

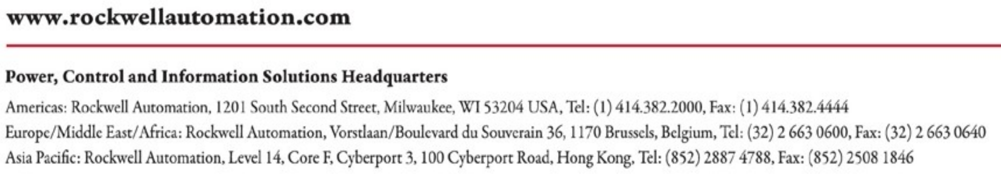
30.4 The remote thin-client shall be an internet browser with a small plug-in, without the need for an additional client software installation.

30.5 The HMI shall support multiple concurrent thick and thin-client connections.

30.6 The HMI shall provide a mobile-only framework, so that the application designer can choose to optimize HMI content by defining a different visual experience for a mobile device. This mobile framework shall allow graphics to be dynamically scaled to fit the device form factor.

30.7 The HMI shall provide a mobile alarm view that can display an alarm table optimized for viewing alarms on mobile devices without requiring an operator station. The alarm table shall display complete alarm details.

30.8 The mobile alarm view shall allow acknowledgement or shelving of alarms from the web client. Access to this feature shall be controlled by security settings configured by the application designer.



www.rockwellautomation.com

Power, Control and Information Solutions Headquarters

Americas: Rockwell Automation, 1201 South Second Street, Milwaukee, WI 53204 USA, Tel: (1) 414.382.2000, Fax: (1) 414.382.4444
Europe/Middle East/Africa: Rockwell Automation, Vorstlaan/Boulevard du Souverain 36, 1170 Brussels, Belgium, Tel: (32) 2 663 0600, Fax: (32) 2 663 0640
Asia Pacific: Rockwell Automation, Level 14, Core F, Cyberport 3, 100 Cyberport Road, Hong Kong, Tel: (852) 2887 4788, Fax: (852) 2508 1846