CRIMSON 3 REVIEWERS' GUIDE

Welcome to Crimson 3—the latest version of Red Lion's widely-acclaimed operator interface configuration software. As you will notice, Crimson 3 contains myriad new features designed to make it easier to design and implement attractive and powerful user interface solutions. This document contains a quick rundown of the new features, and provides a framework for anyone wanting to assess the software. It does assume some level of knowledge of Crimson 2, and does not attempt to detail the features carried over from that earlier version.

USER INTERFACE

- Crimson's new user interface provides rapid access to all areas of the database, and to the various resources used to construct a configuration. The forward and back operations allow Internet-style navigation, making it easy to retrace your steps after editing another area of the database.
- Global undo allows all changes to the database to be reversed, right back to the point at which the file was loaded. Undo and redo are not just limited to the graphics editor or the program editor, but work with every single action performed while editing.
- Universal drag-and-drop allows tags, display pages or any other items to be transferred within databases—and between multiple instances of Crimson. You can open a database, and then drag pages to another database that is open in another copy of the editor.
- Global and local find commands allow text strings and expressions to be located within the entire database, or just within a given page or program. Item find commands allow all items matching a particular name to be visited in turn.
- Usage of ports, devices, tags, program and pages can be tracked, with all items using a particular item being capable of being visited in turn. Database errors can also be located, allowing each to be visited and corrected until all have been removed.
- Tag, display pages and programs can be organized in folders and nested to any depth, allowing databases with many thousands of items to be better organized, and allowing object-oriented tag hierarchies to be constructed.

DATABASE FILES

- Crimson 2 database files can be imported into Crimson 3, with the vast majority of legacy design idioms being supported. The various tag types used by Crimson 2 are automatically mapped to the appropriately-configured Crimson 3 tags, and primitives are likewise converted.
- Databases can be converted from one target device to another, with all page layouts and communications port mappings being adjusted to fit the new hardware.
- Password protection can be applied to database files, locking their contents from either changes, or from unauthorized access.
- An auto-save function saves your work every 15 minutes, and will recovery unsaved work the next time that Crimson is started after a crash or power failure.

COMMUNICATIONS

- Most communications parameters can now be set to expressions, allowing runtime modification of the unit's configuration. Expressions or tags are placed directly in the relevant configuration fields, avoiding the need for special function calls.
- The network infrastructure now supports multiple Ethernet ports, and a user-defined routing table for complex network topologies. All Ethernet configuration options can be set to expressions, allowing the end user to modify IP settings at runtime.
- Additional network protocols can be added beyond the four supported in Crimson 2. Virtual serial ports are also supported, allowing serial protocols to be used over an encapsulated TCP/IP link for use with remote serial servers.
- IP download information is now stored in the database file, allowing easier switching between databases without the need to adjust the download settings, and without the danger of sending the wrong database to the wrong IP address.
- Gateway blocks now allow the selection of either raw tag data, or tag data that has been manipulated according to the tag's scaling properties. This allows more flexibility in system design, and avoids the need for duplicated tags.
- Gateway blocks provide additional properties for controlling their update, allowing a simple enable or disable option as well as Crimson 2's request and acknowledge mechanism. Timed updates are also supported, allowing a block to be processed every so-many seconds.

DATA TAGS

- The number of data tag types has been dramatically reduced, with three basic types being capable of representing all of Crimson 2's many options. A simple tag type is offered for convenience where the richness of the default tags is not needed. Tag labels may be expressions.
- Numeric tags now represent both integer and floating-point data items, and support a large number of manipulation options to ensure all external data can be accessed correctly. Scaling to either integer or floating-point values is also supported.
- Formatting options are decoupled from the tag type, allowing any of seven formatting options to be applied to a numeric value. Format types of numeric, scientific, time and date, two-state, multi-state and IP address can be selected.
- Formatting of a tag can be linked to that of another tag, allowing a folder to be populated with format templates from which other tags can draw. This avoids the duplication of data, and makes it much easier to localize a database for international use.
- Multi-state formats now support expressions for both their data values and the corresponding strings, allowing the behavior of the formatting to be changed at runtime, or to be based upon other data values. Either floating-point or integer values may be used.
- Alarm and event text may now be an expression, allowing dynamically derived alarm names that include, for example, live data. Alarms and events may be set on arrays, allowing the same logic to be applied to each element in the array.

- Flag tags now represent any data item that resolve to an on or off state. The underlying data may be a bit, an integer value or a floating-point value. The state of a flag may be inverted, and multiple bits may be extracted from communications values to create bit arrays.
- String tags support a richer variation of packing types, allowing ANSI or Unicode text to be extracted from a variety of underlying data formats. Byte and character ordering can be adjusted to suit the layout used by the remote device.
- Paste Special and Copy From operations allow properties from one tag to be applied to many tags, with either selected subsets or user-defined lists of properties being transferred. This makes it very easy to edit multiple tags in what amounts to a single operation.
- Smart Duplicate allows a tag to be duplicated, with references to source data being incremented to reference the next element of a comms device, an array or a list of other tags. This allows sets of tag to be created very quickly, without the need for individual editing of the tag properties.

CODE EDITING

- Tags can be created directly from expressions by either selecting the New Tag option or typing in an expression that refers to a non-existent tags. This avoids the need to navigate away from the current field when realizing that a new tag is needed.
- Expression or actions can be edited via the syntax-coloring code editor, allowing complex code fragments to be viewed more easily than is possible within the limited space provided when editing a property directly.
- Any expression or action can be set to a complex code sequence—what is in effect a local program. For expressions, the program must return a data value of the type required by the context, allowing complex operations without the need for global programs.
- Tags, system variables and system functions can be drag-and-dropped into any expressions. Help on system functions is available from the resource pane, or directly from the code editor by pressing F1 after typing the function name.
- Communications ports, comms devices and data logs can be referred to by name in expressions, avoiding the need to find the number of the items as per Crimson 2. Changes made to item names are automatically reflected in any code where they are used.
- Display pages can be created directly from an action by typing code that refers to a non-existent page. As with tags, you will be prompted as to whether you wish to create the item. The user interface element for selecting a page name allows pages to be created.
- Data tags and display pages have properties that can be accessed in expressions and actions by means of the dot operator. For example, Tag1.AsText can be used to access the current value of the specified tag, formatted as a text string.
- All string expressions are now evaluated in Unicode, making it much easier to handle the localization of applications for international deployment. String tags handle conversion to and from 8-bit character sets when dealing with non-Unicode devices.

Pages and Primitives

- Each display page may nominate another page as a master slide, causing the latter to be displayed on the former as a background page. All interactive and live elements from the master slide remain interactive and live on the client pages.
- Color properties may be set to one of 256 or 32768 values depending on the target device. Color animation is no longer based purely on tag colors, but can now be directly configured to flash, switch or smoothly transition between colors based on suitable expressions.
- Most primitives support the addition of text. This text can be edited directly on the page, and will automatically be formatted to multiple lines if required. Text may either be static or based on an expression. Text color may be animated, and drop shadows are supported.
- Multi-line text may include embedded values by means of expressions. The text is automatically reflowed at runtime to fit within the host primitive, allowing complex information to be displayed in an attractive format.
- Most primitives also support the addition of live data. The data may be configured for display only, or for data entry. There are no specific tag data primitives, as any primitive may be configured to display data of any type in any format. Format information may be linked to the source data.
- Actions can be added to all primitives that do not have an inherent action of their own. As with Crimson 2, a variety of actions may be defined, but Crimson 3 now allows the use of local programs to define more complex behaviors without reference to global programs.
- All geometric primitives support tank fills, replacing the simple bars used in Crimson 2. Fills may be applied in any of four directions, and work with polygons, ellipses and other non-rectangular types. Fill colors may be dynamically animated.
- Primitives now support a variety of graduated fills, allowing shading effects to be applied to any figure. These fills work with polygons, ellipses and other non-rectangular types. Fill colors may be dynamically animated. Primitive edges may be several pixels thick.
- A new vertical scale primitive provides automatic placement and spacing of tick marks, with scale labels being placed at appropriate intervals. Scale limits can be dynamically varied, allowing runtime generation of correctly-formatted axis labels.
- Image-based primitives support a property to maintain image aspect during editing. The animated image primitive allows images to be rendered in color or black-and-white based on an expression, providing an easy way to illustrate which user interface elements are disabled.
- A variety of buttons, indicators and toggle switches provide preconfigured access to images in the symbol library. Each variation is provided in a variety of colors, and its behavior can be modified to model a variety of latching, non-latching and biased operations.
- Primitives performing actions may be protected, forcing user confirmation before they can be activated, or requiring the primitive to be unlocked before it can be used. This prevents critical functions from being triggered by accident.

- Popup keypads on the target device may be resized to use much more of the screen for applications where gloves are in use. The alphanumeric keypad supports Pin-Yin entry of Chinese text for international applications.
- Widgets allow the creation of predefined groups of primitives with custom data items and complex predefined behaviors. Automatic binding of widgets to tags and folders allows a form of object-oriented design, with object properties automatically bound to the appropriate data items.

POPUP PAGES

- Popup pages can now contain editable fields, removing the limitation that Crimson 2 imposed on the use of the popup keypad from within a popup page.
- Popup pages can be nested, allowing popups to be displayed on top of other popups to any sensible depth. Actions exist to hide either the topmost popup, or all active popup pages.
- Modal popups allows a popup to be activated from a program, suspending other input activity until the popup is removed by a timeout or by the user providing the requested input. This allows much easier implementation of yes-or-no boxes and similar user interface techniques.

DYNAMIC ANIMATION

- In addition to the movement of images, Crimson 3 allows any primitive or group of primitives to be moved within a specified area based on dynamic values. Movement may either be in the horizontal and vertical directions, or may be specified in terms of an angle and a radius, allowing primitives to orbit a particular location at runtime.
- Items constructed from multiple line segments such as arrows, polygons and stars may be rotated dynamically at runtime. The rotation may be based upon tag values, with limits being specified to allow the rotation effect to be scaled appropriately. Rotating elements may still contain live data or text, and may still contain tank fills.

FONT HANDLING

- A new default font called Hei is available in a variety of sizes and weight, with several sizes supporting full Unicode operation for Chinese and other Asia-Pacific languages. The font is optimized for display of accented characters without prejudice to glyph size.
- A large numeric-only font with anti-aliased characters is provided to allow large digit displays to be constructed without prejudice to display quality. Custom fonts over 32 pixels in height are also anti-aliased on appropriate target platforms.
- User-defined fonts can be modified to include or exclude certain glyphs, allowing better international support while limiting memory usage for those fonts that do not, for example, need to support Chinese characters.
- The Font Manager supports the replacement of a given font with another, allowing the appearance of databases to be modified without the need to edit each individual primitive. Unused fonts can be deleted, and font usage can be examined.

GRAPHICS EDITOR

- The graphics editor has been completely rewritten. It now supports better zooming and panning, drag-and-drop operation between pages and databases, import of just about any possible graphics type, and makes full use of alpha-blending for a clearer user interface.
- The graphics editor now supports editing within groups, allowing items to be moved or edited within a group without that group having to be broken apart. Editing within groups may be nested to any level, allowing multi-level drilldown without disturbing the group structure.
- Tags, programs or system functions may be drag-and-dropped directly from the Resource Pane and placed on a page. Multiple tags or groups of tags may be dropped at once, allowing lists of data values to be created and aligned in a single operation.
- Quick Align allows selected primitives to be aligned to other primitives by selecting the operation and clicking the reference object. Manual alignment is now capable of automatically selecting the reference primitive, avoiding alignment mistakes.
- Paste Special and Copy From operations allow properties from one primitive to be applied to many primitives, with either preselected or user-defined subsets of properties being transferred. This makes it very easy to edit multiple primitives in what amounts to a single operation.
- Smart Duplicate allows a primitive to be duplicated and aligned with the source primitive, with references to source data being incremented to reference the next element of a comms device, an array or a list of tags. This allows tables of values or arrays of buttons to be created very quickly.
- The latest version of Symbol Factory is integrated directly within Crimson, allowing categories and symbols to be viewed straight from the graphics editor. Colorable symbols are pre-colored in a variety of options. Bitmap textures are supported, and use tiled scaling behavior.

IMAGE MANAGER

- The Image Manager allows database images to be reviewed, and allows a given image to be replaced with another at the global level. Unused images can be removed from the database, and images dragged directly into the database can be saved to disk.

LOCALIZATION

- Richer facilities are provided to allow the selection of target languages. Numeric formatting can be configured to follow the selected language, or to stay with the default setting. The Windows keyboard layout can be configured to change automatically during text editing.
- Automatic translation is supported via lexicon files of standard terms from the industrial controls field, or via use of either Google or Microsoft’s web translation API. The entire database or just a single string can be translated in this manner.
- Translations may be exported to or imported from ANSI or Unicode text files for editing in applications such as Microsoft Excel. User-defined lexicons can be generated from existing databases and can be applied globally to new databases.

PROGRAMS

- The new program editor supports syntax coloring, automatic indentation, virtual spaces and automatic help on system functions. Programming errors are better highlighted, and un-translated changes are not lost when a database is saved.

SECURITY

- The addition of a check-before-operate setting to the security descriptor allows the user to be forced to confirm all changes to a particular tag, even if the user has enough rights to perform the adjustment. This prevents critical data values from being changed accidentally.

G3 EMULATOR

- The emulator now supports downloaded drivers, allowing a wider range of communications protocols to be executed on the PC during testing. Multiple instances of the emulator can be active, with the only limit being one of each model of target device.