NetReady™ 3.0 Network Setup Assistance Utility Specification

Document Version 1.1.0

UpdateLogic Incorporated
508-624-8688 (TEL)
508-624-8686 (FAX)
http://www.updatelogic.com

Thank you for using the UpdateLogic NetReady™ 3.0. NetReady provides comprehensive, cost-effective services which allow remote support sessions, software and firmware updates, secure field provisioning of DRM keys, and complete device management for all types of Internet-enabled CE devices.

If you find any problems with this software, please report them via email to support@updatelogic.com.

Please visit our website at www.updatelogic.com for the latest news and information.

UpdateLogic Incorporated

508-624-8688 (TEL)

508-624-8686 (FAX)

Table of Contents

1 Overview

For over 50 years, the television industry has been operating on analog technology. Analog televisions contain little or no computational abilities, and have therefore been sold and marketed much the same as a dishwasher or any other home appliance. With the advent of high definition digital content, low-cost digital circuitry that supports Internet connections, and the ubiquity of streaming media services, the digital age is reaching the most powerful of all home appliances, the device.

The connected devices such as TVs and Blu-ray players will soon compete with the home computer as the most complex device in the home. Software updates, the provisioning of streaming media security credentials, remote support facilities are now all a part of modern connected CE devices. UpdateLogic's NetReady technology provides all of these services.

1.1 Change Tracking

Version Number of This Document	Changes
0.1.0	Initial reviewed version.
1.1.0	Adjusted format and product terminology.

1.2 Scope and Audience

This document is intended for use by Internet connected device manufacturers that desire to provide NetReady Network Setup Assistance Utility Specification compliant behavior on their device. This document defines the required characteristics and describes the library and reference app included in the UpdateLogic Device Agent SDK.

2 Introduction

The Network Setup Assistance Utility capability of NetReady facilitates support technicians helping end users to establish Internet connectivity for their consumer electronic device.

Often, in order to assist with this common task a technician needs to instruct the user on how to find and convey over the phone certain information about the device and its current network configuration, as well as how to run network tests on the device. This process is hindered by the differing terminology, organization, display and activation of these items among different devices from different manufacturers.

The Network Setup Assistance Utility capabilities of NetReady address these issues in the following ways:

1. Network Setup Assistance Utility Specification
    a. Defines common terminology, organization, display and activation of:
        i. Basic device information.
        ii. Device network configuration.
        iii. Device network tests.
2. Network Setup Assistance Utility Library

   a. Provides implementation of related device-independent functionality.
   b. Provides framework for related device-dependent functionality.
3. Network Setup Assistance Utility Reference App
   a. Demonstrates a GUI app compliant with the specification and that uses the library implementation.

3 Specification

3.1 General

3.1.1 Launching

- Sequential button presses on the device remote control of "[Input", "7", "5", "3"] MUST launch the NetReady Network Setup Assistance Utility compliant user interface.
  - A simple and consistent mechanism for launching the user interface simplifies technician instructions and end user actions.

3.1.2 Fields

The following points apply to all of the fields of information described subsequently in this section.

- The fields MUST be listed vertically in the order specified in the tables.
- The fields MUST be named and formatted according to the tables.

3.2 Basic Device Information

Basic information about the device that technicians must often solicit from the end user is clearly displayed in a consistent manner.

3.2.1 Fields

- The fields MUST be in the left-most column of the display.
  - Correlates to the information being the first solicited by a technician.
  - Makes it easy for the technician to describe to the end user where to find the information.
- The fields MUST be visually grouped together independently from other information on the display.
  - Helps the end user focus on a subset of all of the information.
- The grouping SHOULD be distinguished visually from other groupings on the display in terms of background, color and/or format.
  - Once conveyed to the technician, this information is not part of the "working set" of fields during network troubleshooting, and it cannot be edited/activated. Its unique "read-only" and "used once" use pattern can be conveyed visually to prevent the end user from trying to click on or edit the information.

Field Name	Description	Example Value
Make	Name of device manufacturer or brand.	Acme
Serial Number	Device serial number, preferably with all letters capitalized.	123ABDA552712
Model	Name of device model, preferably with all letters capitalized.	HTP-612
Software Version	Version of device's current software, preferably with all letters capitalized.	1.78.2.14A
For Support Call	Technical support phone number for the device.	1-800-555-1212

Table 1: Basic Device Information Fields

3.3 Device Network Configuration

Information about the device network configuration that technicians must often solicit from the end user is clearly displayed in a consistent manner.

3.3.1 Fields

- The fields MUST be in the second-left-most column of the display.
    - Correlates to the information being the second solicited by a technician.
    - Makes it easy for the technician to describe to the end user where to find the information.
- The fields MUST be visually grouped together independently from other information on the display.
    - Helps the end user focus on a subset of all of the information.
- The grouping MUST have a clear "Network" heading.
    - Aids in identifying the correct location of requested information.
- The field names SHOULD be right justified and the field values SHOULD be left justified.
    - Allows for easier correlation of name/value pairs from a distance.
- The fields of each network interface MUST appear as a sub-group named for the network interface "Name" field.
    - Associates the fields of a network interface together.
- Each network interface sub-group SHOULD appear as a set of fields under a left justified visually distinct header row using the name of the network interface.
    - Separates network interface sub-groups while retaining a flat and simple display not requiring end user navigation.

- The entire set of fields in *Table 1: Basic Device Information Fields* MUST appear for each enabled network interface on the device.
    - A consistent set of network interface information prevents miscommunication.
- Each disabled network interface on the device MUST show either the entire set of fields in *Table 1: Basic Device Information Fields* OR MUST show exactly the "Type" and "Status" fields.
    - Not all information about disabled network interfaces may be available.
- Editing for the "Network Name" field for wireless network interfaces SHOULD be implemented as a drop-down list of discovered wireless networks that also shows the signal strength for each option.

Field Name	Description	Example Value	Editable
Name	Name.	"Wired", "Ethernet", "eth-0", etc…	No.
Type	Type as "Wired" or "Wireless".	Wireless	No.
Status	Status as "Disabled", "Connected" or "Disconnected".	Connected	Yes, set to "Enabled" or "Disabled".
Network Name	For wireless interfaces only, the name (SSID) of the wireless network set for this interface.	MyNet	Yes, text field.
Security Key	For wireless interfaces only, the wireless network security key set for this interface.	theFamilyNet!	Yes, text field.
DHCP	DHCP setting as "Enabled" or "Disabled".	Enabled	Yes, set to "Enabled" or "Disabled".
IP Address	IP address.	192.168.2.172	Yes, if "DHCP" is "Disabled".
Gateway	Gateway IP address.	192.168.2.1	Yes, if "DHCP" is "Disabled".
Subnet Mask	Subnet mask.	255.255.255.0	Yes, if "DHCP" is "Disabled".
Physical Address	Physical (a.k.a., MAC or hardware) address with all letters capitalized.	AA:BB:CC:DD:EE:FF	No.
First DNS Server	First DNS server IP address.	192.168.2.1	Yes, if "DHCP" is "Disabled".

Second DNS Server	Second DNS server IP address.	68.87.72.134	Yes, if “DHCP” is “Disabled”.
Third DNS Server	Third DNS server IP address.	68.87.72.135	Yes, if “DHCP” is “Disabled”.

Table 2: Network Interface Fields

3.3.2 Error Fields Values

- Any field value deemed erroneous SHOULD be marked with a red circle “X” icon to the right of the field value.
    - A consistent and distinct error indicator helps a technician easily ask the end user about any errors being displayed.
    - Examples of erroneous field values:
        - Interface “Status” is “Disconnected”.
        - Wireless “Network Name” is not reachable.
        - Wireless “Security Key” is rejected by the wireless network.
        - DHCP is “Enabled” but no DHCP server was found.
        - Statically assigned “IP Address”, “Gateway” and “Subnet Mask” are inconsistent (for example, the “Gateway” is not on the same subnet as the “IP Address”).
        - Missing “DNS Server” values when “DHCP” is “Enabled”.

3.3.3 Buttons

Two buttons appear at the bottom of the group.

- The “Reset” button MUST reset all network interfaces and, after an implementation dependent delay to allow for DHCP renewal and similar, run all of the network tests (see section **Error! Reference source not found. Error! Reference source not found.**).
    - This is a common initial troubleshooting task.
- The “Accept” button MUST commit all changed field values to the device.

3.4 Device Network Tests

Simple tests useful in network troubleshooting are clearly displayed and activated in a consistent manner.

3.4.1 Tests

- The tests MUST be in the third-left-most column of the display.
    - Correlates to the tests typically being the third used by a technician.
    - Makes it easy for the technician to describe to the end user where to find the tests.
- The tests MUST be visually grouped together independently from other information on the display.

- Helps the end user focus on a subset of all of the information.

- The grouping MUST have a clear "Tests" heading.
  - Aids in identifying the correct location of the tests.
- The test names SHOULD be right justified and the test results SHOULD be left justified.
  - Allows for easier correlation of name/value pairs from a distance.

Test Name	Description	Result
Local Ping	Ping (round-trip ICMP) to the gateways of each connected network interface.	"Pass" or "Fail".
Internet Ping	Ping (round-trip ICMP) to a high-availability Internet server.	"Pass" or "Fail".
DNS Lookup	Resolve a well-known Internet site name.	"Pass" or "Fail".
Speed (Up)	Measure sustained Internet upload speed.	Numeric value in megabytes-per-second to one decimal place (i.e., "1.2").
Speed (Down)	Measure sustained Internet download speed.	Numeric value in megabytes-per-second to one decimal place (i.e., "8.5").
Latency	Measure average round-trip packet latency to a high-availability Internet server.	Numeric value in milliseconds (i.e., "550").
Wireless Signal	For primary connected wireless interface only, the strength of the wireless network signal.	Numeric value in dBm (i.e., "-42").
Wireless Speed	For primary connected wireless interface only, the negotiated wireless network data rate.	Numeric value in megabytes-per-second (i.e., "54").

Table 3: Device Network Tests

3.4.2 Error Test Results

- Any test result deemed erroneous SHOULD be clearly denoted with a subtle red outline and highlight.
  - A consistent and distinct error indicator helps a technician easily ask the end user about any errors being displayed.

3.4.3 Buttons

One button appears at the bottom of the group.

- The "Run" button MUST cause all device network tests to execute serially and for

their results to be displayed.

4 Library

The UpdateLogic Device Agent contains routines useful in implementing this specification. The reference implementations of adaptation functions and external APIs provide the infrastructure and example handling which can be modified for device specific handling.

4.1 Adaptation Functions

`void UtvCEMLocalSupportInitialize( void )`

This function is used to handle platform specific initialization for the Local Support feature. The reference implementation registers command handlers with the embedded web server for the Local Support commands.

NOTE: The embedded web server is an optional feature of the UpdateLogic Device Agent.

`UTV_RESULT UtvCEMGetDeviceInfo ( cJSON * pDeviceInfo )`

This function is used to handle platform specific retrieval of device information. The reference implementation adds the specification required values to the JSON object.

`UTV_RESULT UtvCEMGetNetworkInfo ( cJSON * pNetworkInfo )`

This function is used to handle platform specific retrieval of network information. The reference implementation adds the specification required values to the JSON object.

Review the implementation of the platform adaptation function, UtvPlatformGetNetworkInfo, and if necessary replace the call to that function with device specific processing to retrieve the expected network information and add the specification required values to the JSON object.

`UTV_RESULT UtvCEMNetworkDiagnostics ( cJSON * pNetworkDiagnostics )`

This function is used to handle platform specific execution of network diagnostics and return appropriate results. The reference implementation adds the specification required values to the JSON object.

`UTV_RESULT UtvPlatformGetNetworkInfo ( void * pVoidResponseData )`

This function is used to handle platform specific retrieval of network information. The reference implementation adds the specification required values to the JSON object using Linux network functionality.

NOTE: The pVoidResonse is a pointer to a cJSON object.

4.2 External APIs

`UTV_RESULT UtvProjectOnLocalSupportGetDeviceInfo ( UTV_BYTE ** ppszDeviceInfo )`

This function is used to retrieve the device information as a JSON formatted string. This function should not require modification.

NOTE: The UpdateLogic Device Agent code allocates storage for the JSON formatted string. The calling application must free this memory.

UTV_RESULT UtvProjectOnLocalSupportGetNetworkInfo (UTV_BYTE ** ppszNetworkInfo)

This function is used to retrieve the network information as a JSON formatted string. This function should not require modification.

NOTE: The UpdateLogic Device Agent code allocates storage for the JSON formatted string. The calling application must free this memory.

UTV_RESULT UtvProjectOnLocalSupportNetworkDiagnostics (UTV_BYTE ** ppszNetworkDiagnostics)

This function is used to execute the network diagnostics and retrieve the results as a JSON formatted string. This function should not require modification.

NOTE: The UpdateLogic Device Agent code allocates storage for the JSON formatted string. The calling application must free this memory.

5 Reference App

The UpdateLogic Device Agent SDK contains a Yahoo Connected TV Widget reference app.

The app contains a set of JavaScript files and graphical images that demonstrate user facing functionality of the NetReady Network Assistance APIs and the Live Support APIs. Emphasis was put more on the demonstration of the APIs over production readiness so some work may be required if the reference app is productized.

The reference application uses the optional Device Agent embedded web server to cross the widget to Device Agent application boundary. This implementation could be replaced by device specific code that interacts with the Device Agent through the C functional API.

> The Device Agent embedded web server is an optional feature that is used for development and test. This is not for production use at this time. Please contact UpdateLogic for questions or additional information regarding this functionality.

Note: Any JavaScript or graphical images (with the exception of the NetReady branding) included with the reference app can be reused as required. Adobe Photoshop and/or Adobe Fireworks files are available upon request.

5.1 Screen Shots

It is not the intention to define an integrators brand experience. The screenshots are provided as a reference design of 2 potential implementations.

The following screenshots demonstrate how an integrator's design department can create a design that provides the needed layout and flow while at the same time creating a unique brand experience.

Figure 1. Actual Reference implementation

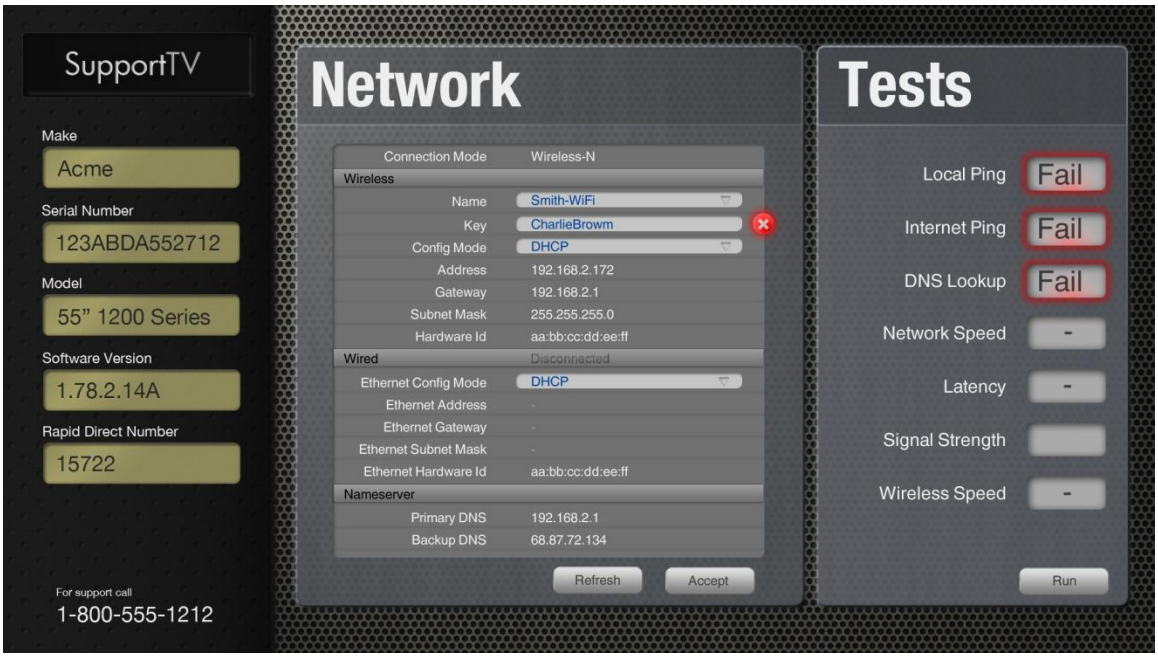

Figure 2. Alternate Reference design example.

5.2 Reference App Code Layout

The reference app top level directory structure is laid out as follows:

- Contents

- Images – Contains all images used in the reference app.
- Javascript – Contains the actual code used in the reference app.
- widget.xml – Contains the Yahoo Widget XML app definition. This needs to be updated with the relevant integrator information.
- main.TV- Yahoo widget boilerplate

5.2.1 Images

The images directory contains a single subdirectory, 960x540, which contains the relevant images used in the application.

5.2.2 JavaScript

The JavaScript directory is broken up into several different directories and files.

- core
  - EventHandlers.js – Contains event handlers that are core to the application as a whole.
- model
  - LocalAgentInterface.js – Contains the actual loopback network calls made to the agent. Additionally this contains some basic processing of the returned data.
- views
  - NetConnectSidebarView.js – Contains the initial sidebar view. This is the first view that is loaded when the user clicks the widget in the dock. The initial calls to the LocalAgentInterface to fetch device info, network info, and run the network tests are located in this view.
  - NetConnectStartView.js – Contains the view that is loaded when a user initiates a Live Support session.
  - NetTestFullscreen.js – Contains the Device Info, Network Info, and Network Test fullscreen view.
  - NetTestSelectListView.js – Contains a sidebar view that is used as a “Combo” select control. This is spawned from the NetTestFullscreen view.
  - SnippetView.js – Contains the widget dock control (this is viewable when a user brings up the Widget Dock).
- init.js – The main entry point into the widget.