On-Device troubleshooting for conferencing and enterprise equipment

Micah Lemonik
Sujit Varghese
Barine Tee

Follow this and additional works at: http://www.tdcommons.org/dpubs_series

Recommended Citation
Lemonik, Micah; Varghese, Sujit; and Tee, Barine, "On-Device troubleshooting for conferencing and enterprise equipment", Technical Disclosure Commons, (November 13, 2017)
http://www.tdcommons.org/dpubs_series/806

This work is licensed under a Creative Commons Attribution 4.0 License.
This Article is brought to you for free and open access by Technical Disclosure Commons. It has been accepted for inclusion in Defensive Publications Series by an authorized administrator of Technical Disclosure Commons.
On-Device troubleshooting for conferencing and enterprise equipment

ABSTRACT

This disclosure describes techniques for largely automated on-device troubleshooting of conferencing and enterprise equipment through a simple user interface. Through the interface, troubleshooting can be performed for peripherals such as cameras, speakers, microphones, touch controllers, and near-field communication (NFC) devices. In case of a test failure, additional logging information is captured and sent for further processing to determine if the issue can be resolved using a software fix, or alternatively, if a hardware fix is necessary. On-device troubleshooting provides a local, offline interface to diagnose problems with equipment and reduce escalations to next level of support and/or returns of the equipment or parts.

KEYWORDS

- On-device troubleshooting
- Videoconferencing system
- Enterprise hardware
- Interactive whiteboard

BACKGROUND

When enterprise hardware equipment, e.g., a videoconferencing system malfunctions, it is difficult for users to isolate the reason as a hardware failure or a software issue. In some instances, users choose to generate a support ticket, and return hardware parts or the entire system, even when the source of the problem is in software. Part or whole system returns, e.g., via a return merchandize authorization (RMA) process can be expensive for hardware vendors.
DESCRIPTION

Techniques described provide an on-device offline interface to troubleshoot problems with enterprise equipment such as videoconferencing systems, interactive whiteboards, etc. In addition to troubleshooting device peripherals such e.g., camera, microphone, speaker, touch controller, etc. the techniques permit capture of device logs that can be sent for further analysis. Installation and management of enterprise wireless certificates from external devices can also be performed using the on-device offline interface.

The techniques incorporate the functionality of a third-party verification or validation software but additionally streamline and automate troubleshooting via an on-device user interface. To initiate the on-device user interface, a device or module, e.g., USB module, is connected to a corresponding port the conferencing equipment. The conferencing equipment recognizes the module, e.g., based on a USB registration event, and displays a user interface for authentication. User access is verified using a rolling password, e.g., based in part on an over-the-air (OTA) image or build fingerprint of the conferencing equipment. The user interface for troubleshooting is displayed only after the user has successfully completed authentication.

Device peripherals such as the camera, microphone, speakers, NFC, and touch controller are validated using different automated troubleshooting tests. Each successful test indicates a respective functional peripheral device. The user interface enables users to view test runs and corresponding via a troubleshooting console. When a test fails, the relevant logs are provided, e.g., sent for analysis, to determine whether a software fix can resolve the issue. The user interface also enables viewing of artifacts used for troubleshooting, e.g., images utilized to test a camera peripheral functionality.
Failure of a peripheral can occur due to non-functional firmware and/or non-responsive hardware. In case of non-functional firmware, a software update, patch, or reflash is sufficient to resolve the problem. However, if the test indicates non-responsive hardware and if a software update is ineffective, the particular peripheral or equipment is identified as suitable for hardware replacement via a return merchandise authorization process.

Fig. 1: On-device troubleshooting user interface

Fig. 1 illustrates an example user interface for on-device troubleshooting of enterprise equipment. A screen of the conferencing equipment (100) displays the troubleshooting user interface (102). The user interface includes a module selection panel (104) that permits selection of the particular module of conferencing equipment for test. In the example illustrated in Fig. 1, the camera is under test. Log information from the test is displayed on a second panel (106). A third panel displays artifacts used during the test run (108).
**Camera**

Troubleshooting camera functionality involves determining whether the camera is able to detect signals, e.g., by sampling texture feeds. The camera texture feed is sampled over multiple, e.g., three, texture update periods and tested for variation. Variations across the three periods over random sample point groups indicate that the camera is able to detect a signal. If the camera is unable to detect a signal, the entire surface area is detected as black with no variation across samples. This technique specifically tests the ability of the camera hardware to detect a signal, and does not test for orientation or focus.

**Speakers**

A speaker test is performed by using a tone generator is used that broadcasts a tone periodically, e.g., once every second. The speakers are first tested at a high volume, e.g., full volume, by playing the test tone. The test tone is of an amplitude and frequency that can be sampled by the device microphone (if it is functional). A feedback loop is used to detect the amplitude of the broadcasted tone and to ensure that the tone passes a minimum threshold over a number of successive intervals. If the tone is sampled successfully, the speaker test is deemed a success. A successful speaker test also indicates a functional microphone since only a working microphone can properly sample and verify the sound output of speakers.

If no tone is detected over a number of intervals, the speaker is deemed to have failed the automated test. However, since the troubleshooting utilizes a microphone module to verify the speaker output, it is possible that the failure is due to a defective microphone. To prevent a false negative result, a prompt is provided for the user to provide confirmation of speaker output. If user confirmation is obtained that a tone from the speaker is audible, the speaker test is deemed successful, and the microphone is detected as defective. Else, if no user confirmation is obtained,
e.g., when the user cannot hear the tone from the speaker, the respective speaker is considered defective.

**NFC**

The first level of troubleshooting verifies whether the NFC device is available and enabled. If the NFC is unavailable, the test immediately fails. However, if the NFC is disabled, the user is prompted to enable it. The second level of troubleshooting, once the NFC is both available and enabled, includes sending and receiving an NFC message. An NFC tag may be used. For example, a smartphone based NFC app is used to perform NFC test.

**Microphone**

A tone generator and ambient noise are used to troubleshoot the microphone. It is determined whether the microphone sampled any valid data. Since a defective microphone module is unable to sample any sound, the amplitude threshold of the test is set at a low value. In this case, ambient noise triggers a spike that confirms that the microphone is functional. Additionally, if noise spikes, including the generated tones, are registered for a number of successive intervals, the microphone is deemed functional. Alternatively, if no ambient noise or tones is detected over a number of intervals, then the test is deemed a failure. The recorded amplitude is displayed on the troubleshooting user interface.

**Touch controller**

The automated test for a touch controller determines only whether the touch hardware is functional and not whether touch is detectable on the screen. If touch is not detected on the screen, the user interface for troubleshooting cannot be used directly. In this case, an additional peripheral such as a mouse is required to be connected to the equipment to perform the
troubleshooting. To determine whether a touch sensor is functional, sensor statistics are analyzed. If no sensor statistics are available, the test is deemed a failure. When statistics are retrieved, the sensor status is verified (e.g., “OK”) and it is confirmed whether the sensor data rate meets a threshold. Further, if the temperature of the CPU of the equipment is not within tolerance, the test is deemed to fail. Further, it is verified that the thermal management control unit does not constantly reset. Such constant resetting is an indicator of the sensor activating due to software issues such as kernel panic or software crash, and can cause the touch recognition to be sporadic at best. Test results and corresponding logs are displayed on the troubleshooting user interface.

CONCLUSION

This disclosure describes techniques for largely automated on-device troubleshooting of conferencing and enterprise equipment through a simple user interface. Through the interface, troubleshooting can be performed for peripherals such as cameras, speakers, microphones, touch controllers, and near-field communication (NFC) devices. In case of a test failure, additional logging information is captured and sent for further processing to determine if the issue can be resolved using a software fix, or alternatively, if a hardware fix is necessary. On-device troubleshooting provides a local, offline interface to diagnose problems with equipment and reduce escalations to next level of support and/or returns of the equipment or parts.