SYSTEM TO AUTOMATICALLY PROCESS COMPONENTS ON A DEVICE

Cross-Reference o Related Applications

[0001] This Application claims the benefit of priority of a pending U.S. Provisional Patent Application entitled "AUTO-INSTALL" having Application No. 60/435,486 and filed on December 20, 2002, the disclosure of which is incorporated by reference herein in its entirety for all purposes.

[0002] This Application also claims the benefit of priority of a pending U.S. Provisional Patent Application entitled "REGISTRY-BASED AUTO INSTALL COMPONENT HANDLING" having Application No. 60/435,828 and filed on December 20, 2002, the disclosure of which is incorporated by reference herein in its entirety for all purposes.

BACKGROUND

I. FIELD

[0003] The present invention relates generally to the processing of components on a device, and more particularly, to a system for automatically installing, updating, or removing components on a device.

II. DESCRIPTION OF THE RELATED ART

[0004] Data networks provide a way for a large numbers of users to communicate with each other using a variety of network-enabled devices. For example, in a wireless communication network, a variety of network-enabled portable telephones allow users to communicate with each other over great distances. The network-enabled devices are generally configured with a variety of installed components that control how the device operates, and ultimately, how well the overall network performs.

[0005] In certain circumstances a network operator would like to upgrade, install, delete, or otherwise change the configuration of the network-enabled devices. For example, as system software is improved, the network operator would like to have new components installed on all devices in the network so that the network operates more efficiently. For example, the network operator may like to install application software, a binary executable, or other information on the devices to provide service

improvements or upgrades. In another situation, the network operator may desire to install enterprise applications or other device upgrade applications based on the needs of a specific type of device. Furthermore, if it is determined that a problem exists with a currently installed system component, the network operator would like to install an upgraded component to repair the problem, regardless of whether the device user is even aware that the problem exists. For example, if a problem exists with a current device component that allows a user to view multimedia content, the network operator would like to upgrade all the devices on the network to replace this component with a new component that does not have the problem.

[0006] One technique used to install, upgrade, delete, or otherwise change the components on a device is to wait for the device user to return the device to a repair center. Once at the repair center, repair personnel can reprogram the device so that the device has the most recent component versions. Unfortunately, this process is very inefficient because device users may fail to return the device if they are unaware of the problem, or if the problem is not currently affecting how they used the device. Because not all of the devices will be upgraded, some devices will not operate to provide the best performance and the overall operation of the network may be degraded.

[0007] Therefore, what is needed is a system to automatically process components on a device to allow selected versions of components to be installed and activated. The system should be flexible enough to process the components on a large number of devices in a relatively short time, thereby providing fast upgrades to all devices operating on a network, which will result in the best device performance and increased network efficiency.

SUMMARY

[0008] In one or more embodiments, a system is provided to automatically process components on a device. For example, in one embodiment, the system allows a device to install, update, delete, activate, disable, recall or otherwise change the state of a component on a device using a versioned action list available on a download server. The components that can be processed may be of any type, for example, an application, executable, configuration information, user interface settings, random data, or any other type of information.

[0009] During operation of the system, the device checks the version of the action list on the download server against a stored version associated with the last action list processed by the device. If the two versions are different, the device downloads the new action list from the server and parses each item in the action list to process components on the device. In one embodiment, each item in the action list comprises a component/action pair that associates a component identifier with an action identifier. The component identifier identifies a type of component and its current version. The action identifier identifies an action to be performed by the device with respect to the identified component.

[0010] In one embodiment, the device parses each component/action pair in the action list to determine whether an action needs to be performed for the identified component. The device compares the version of the component in the action list to a component version stored on the device. If the two component versions are the same, the device takes no action with regards to that component/action pair. If the versions are different, the device performs the action associated with the component in the action list. For example, if the action is to install the identified component, the device downloads the component, and any other necessary information from the download server, and installs the component on the device. Thus, the device steps through the action list performing the designated actions only on new component versions.

[0011] After processing a particular component/action pair, the component version stored on the device is updated with the new component version provided in the action list. When the entire action list has been processed, the device records the version identifier of the action list, so as to avoid re-processing the current action list in the future. Thus, the device will not process another action list until the version of the action list available on the download server is different from the stored version on the device.

[0012] In one embodiment, the device checks the version of the action list every time the device communicates with the download server. For example, the device may communicate with the download server after the device is powered on or at periodic intervals. The system provides a mechanism to allow components to be processed on the device with no user interaction or limited user interaction. For example, in one embodiment, components may be pushed to a device at power up to effectively provide

a "silent installation." In another embodiment, a user interface is provided so that the component processing may be at the option of the user. Thus, the system allows the component processing to be forced, prompted, required, or optional.

[0013] The system may be used to process components on a single device or on a large number of devices. For example, in a data network where are large number of devices can access an action list server, a single action list can be delivered to all devices and each device can determine what components to process for that device. In another embodiment, the server may provide multiple action lists that can be used for different device types. For example, different types of devices may access a different action list to process components for that type of device. Thus, it is possible for the system to provide global updates to a large number of devices, where the updates are performed over a period of hours or days as each device contacts the action list server. Additionally, communication between the server and a device can be performed using any type of secure communication technique, such as encryption or any type of encoding, so that the devices can be authenticated and any transmission of information is done in a secure fashion.

[0014] In one embodiment, a method is provided for automatically processing components on a device. The method comprises receiving a version identifier associated with an action list, determining that the version identifier is not equivalent to a stored version identifier, receiving the action list, parsing the action list to obtain a component identifier and an associated action, and performing the action on a component identified by the component identifier.

[0015] In one embodiment, apparatus is provided for automatically processing components on a device. The apparatus comprises logic to receive a version identifier associated with an action list, and logic to compare the version identifier to a stored version identifier. The apparatus also comprises logic to receive the action list if the version identifier is not equivalent to the stored version identifier, and logic to parse the action list to obtain a component identifier and an associated action. The apparatus also comprises logic to perform the action on a component identified by the component identifier.

[0016] In one embodiment, apparatus is provided for automatically processing components on a device. The apparatus comprises means for receiving a version

identifier associated with an action list, and means for determining that the version identifier is not equivalent to a stored version identifier. The apparatus also comprises means for receiving the action list, and means for parsing the action list to obtain a component identifier and an associated action. The apparatus also comprises means for performing the action on a component identified by the component identifier.

[0017] In one embodiment, a computer-readable media is provided comprising instructions, which when executed by processing logic in a device, operate to automatically process components on the device. The computer-readable media comprises instructions for receiving a version identifier associated with an action list, and instructions for determining that the version identifier is not equivalent to a stored version identifier. The computer-readable media also comprises instructions for receiving the action list, and instructions for parsing the action list to obtain a component identifier and an associated action. The computer-readable media also comprises instructions for performing the action on a component identified by the component identifier.

[0018] Other aspects, advantages, and features of the present invention will become apparent after review of the hereinafter set forth Brief Description of the Drawings, Detailed Description of the Invention, and the Claims.

BRIEF DESCRIPTION OF THE DRAWINGS

[0019] The foregoing aspects and the attendant advantages of the embodiments described herein will become more readily apparent by reference to the following detailed description when taken in conjunction with the accompanying drawings wherein:

[0020] **FIG. 1** shows one embodiment of a system to automatically process components on a device;

[0021] **FIG. 2** shows a functional diagram of a device that includes one embodiment of a system to automatically process components;

[0022] **FIG. 3** shows one embodiment of a method for operating a device to provide a system to automatically process components on the device;

[0023] **FIG. 4** shows transactions that occur between a download server and a device during operation of one embodiment of a system to automatically process components on the device; and

[0024] **FIG. 5** shows one embodiment of an action list for use in a system to automatically process components on a device.

DETAILED DESCRIPTION

[0025] The following detailed description describes embodiments of a system to automatically process components on a device. The system is suitable for use in any type of wired or wireless network, including but not limited to, communication networks, public networks, such as the Internet, private networks, such as virtual private networks (VPN), local area networks, wide area networks, long haul network, or any other type of data network. The system is also suitable for use with any type of device that is capable of downloading and installing components. For example, the system is suitable for use with office computers, notebook computers, and handheld devices, such as portable telephones, PDAs, or any other type of device capable of receiving and installing components.

[0026] In one or more embodiments, the system interacts with a runtime environment executing on the device that is used to simplify operation of the device, such as by providing generalized calls for device specific resources. One such runtime environment is the Binary Runtime Environment for Wireless™ (BREW™) software platform developed by QUALCOMM, Inc., of San Diego, California. In the following description, it will be assumed that the device is executing a runtime environment, such as the BREW software platform. However, one or more embodiments of the system are suitable for use with other types of runtime environments to automatically process components on a variety of wired and wireless devices.

[0027] **FIG. 1** shows one embodiment of a system **100** to automatically process components on a device. The system **100** comprises a server **102**, a data network **104**, and a device **106**. The data network **104** may be any type of wired or wireless network that allows information to be communicated between the server **102** and the device **106**. For example, the network **104** may be a communication network, wide area network, virtual private network, or a public network, such as the Internet.

[0028] In one or more embodiments, the system operates to process components on the device **106**. For example, the server **102** includes components **112**, an action list **110**, and an action list version identifier **108**. The version identifier **108** identifies the current version of the action list **110**. The action list **110** comprises component/action pairs where each pair comprises a component identifier and an action identifier. The component identifier is a unique item ID that identifies a particular component. A portion of the component identifier is used to identify a version of the component. For example, the component identifier is a series of bits where a portion of the bits is used to identify the version of the component. The action identifier identifies a particular action, such as install, update, delete, recall, disable, or any other action that the device **106** will perform with regards to the identified component.

[0029] During operation, the server **102** transmits the action list version identifier **108** to the device **106** via the network **104**. For example, the device **106** may contact the server **102** during a power up sequence and the server **102** responds by transmitting the action list version identifier **108** to the device **106**. The device **106** compares the version identifier **108** with a stored version identifier **114** that represents the version of the last action list to be processed by the device **106**. If the downloaded version **108** and the stored version **114** are equivalent, then the device does not need to perform any installations or changes to the existing components **116** on the device. However, if the downloaded version **108** is different from the stored version **114**, then the device **106** knows that additional component processing is required. For example, there may be new components located at the server **102** that need to be installed on the device **106**.

[0030] Once the device **106** detects that a new version of the action list exists, it transmits a request to the server **102** to obtain the new action list **110**. The server **102** responds by transmitting the action list **110** to the device **106**. The device **106** then processes the action list **110** to install, update, delete or otherwise process components identified in the action list **110**.

[0031] The device **106** operates to process the action list **110** by stepping through each component/action pair and determining whether or not to take action with regard to each component. For example, if the action list specifies that a component needs to be installed or updated, the device **106** downloads the component **112** and any other required files from the server **102** and installs it on the device **106**. For example,

information from the server **102** transmitted to the device **106** is shown generally at **120**. If the action list **110** specifies that a component needs to be deleted, the device **106** deletes the component. Thus, each component/action pair in the action list **110** is processed by the device **106** to automatically install, update, delete, etc., the identified component. The action list may be of any length and after processing the action list, the device **106** updates the stored action list version identifier **114** to store the version of the most recently processed action list (i.e., version **108**).

[0032] In one embodiment, the device **106** parses the component/action pairs in the action list **110** to determine whether or not action related to a particular component needs to be taken. For example, the device **106** may already have the newest version of a component installed, and so it is not necessary for the device to re-install that component. In one embodiment, the component identifier in the action list **110** includes information to determine the version of the component. For example, the version information can be appended to the end of the component identifier. The device **106** compares a stored component version **118** to the component version downloaded in the action list **110**. If the stored and downloaded component versions are the same, then the device need not take any action with regards to that component. However, if the stored and downloaded component versions are different, then the device **106** performs the action associated with that component in the action list. Thus, the system is very efficient, since the device **106** only processes new versions of the components.

[0033] **FIG. 2** shows a functional diagram of a device **200** that includes one embodiment of a system to automatically process components on the device **200**. The device comprises processing logic **208**, compare logic **206**, function logic **214**, version update logic **224**, a stored action list version identifier **204**, component version identifiers **212**, and installed components **226**. The described logic and functions provided by the device **200** may be implemented in hardware, software, or a combination of hardware and software. For example, in one or more embodiments, the processing logic **208** comprises a CPU, processor, gate array, hardware logic, memory elements, virtual machine, software, and/or any combination of hardware and software. Thus, the processing logic **208** generally comprises logic to execute machine-readable instructions to perform the functions described herein. It should be noted that the device

200 illustrates just one embodiment and that changes, additions, or rearrangements of the device elements may be made without deviating from the scope of the invention.

[0034] **FIG. 3** shows one embodiment of a method **300** for operating a device, such as device **200**, to provide a system to automatically process components on the device **200**. For the purposes of clarity, the method **300** will be described with reference to the device **200** shown in **FIG. 2**. It will further be assumed that the device **200** is in secure communication with a download server via a data network, as illustrated in **FIG. 1**.

[0035] At block **302**, the device obtains an action list version identifier from the download server. For example, the device communicates with the download server via a data network and the download server transmits the action list version identifier to the device, as shown at **202**.

[0036] At block **304**, a test is performed to determine if the downloaded action list version identifier is different from a stored version identifier that is associated with the last action list to be processed by the device. For example, the downloaded version identifier **202** and the stored version identifier **204** are input to compare logic **206** that compares the two identifiers to determine if they are equivalent. If the two version identifiers are equivalent, the method returns to block **302** to obtain a new version of the action list at another time. If the two version identifiers are different (Diff), the method proceeds to block **306**.

[0037] At block **306**, the device retrieves the action list from the download server. For example, the action list **210** is downloaded from the server to the processing logic **208** via the data network.

[0038] At block **308**, the device begins processing the action list by parsing the first component/action pair in the action list. For example, the processing logic **208** operates to process the downloaded action list **210** to parse the component/action pairs.

[0039] At block **310**, a test is performed to determine what action, if any, is required for the component/action pair that is currently being processed. In one embodiment, the device operates to automatically perform the action by proceeding to block **312**. However, this may result in existing components being re-installed on the device. In another embodiment, the version of the component is checked to determine if the action is necessary. For example, if the action is to "install" the component, the version of the component is checked to see if the device has that version of the component currently

installed. Thus, the method operates to avoid re-installing components that are already installed on the device. For example, the processing logic **208** retrieves a stored component version identifier **212** and compares it to the version of the component identified in the action list. In one embodiment, the version of the component is incorporated in the component identifier provided in the action list. If the two component versions are the same, no further action is required with regards to that component and the method proceeds to block **416**. If the two component versions are different, then the processing logic **208** operates to perform the action associated with the component and the method proceeds to block **312**.

[0040] At block **312**, the action associated with the component in the current component/action pair is performed to change the state of the identified component. For example, the processing logic **208** operates to control the action logic **214** to perform the action of installing, updating, deleting, activating, disabling, recalling or otherwise changing the state of the identified component. For example, a soft recall may be performed where the component is deleted from the device but associated data and/or licensing information is not removed. For example, if the action is to install or update the component, the processing logic **208** operates to download the component **222** (or update) from the download server via the data network. The downloaded component is then installed as an installed component **226**. The processing logic **208** may perform any type of installation or update procedure to install or update the downloaded component **222** as an installed component **226**. If the action is to delete a component, the processing logic **208** controls the delete logic **220** to delete the identified component from the installed components **226**. Although not shown in **FIGS. 2** and **3**, virtually any type of action may be performed with regards to the component, such as installing, updating, deleting, recalling, activating, and deactivating, etc.

[0041] At block **314**, a component version list is updated to reflect that a new version of the component has been installed or updated, or that the component has been deleted. For example, the processing logic **208** controls the version update logic **224** to update the stored component version identifiers **212** with the new information about the currently processed component.

[0042] At block **316**, a test is performed to determine if all of the component/action pairs in the action list have been processed. If all pairs have been processed, the method

proceeds to block **320**. If all pairs have not been processed, the method proceeds to block **318** where the next pair is accessed for processing at block **310**. The action list **210** may be any length and so there may exist any number of component/action pairs to be processed.

[0043] At block **320**, the stored action list version identifier at the device is updated. For example, the processing logic **208** controls the update version logic **224** to update the stored action list version identifier **204** with the identifier associated with the most recently processed action list. Thus, the system will not operate to process another action list until a new version of the action list is available.

[0044] In one embodiment, the system for automatically processing components on the device comprises program instructions stored on a computer-readable media, which when executed by the processing logic **208**, provides the functions described herein. For example, instructions may be loaded into the device **200** from a computer-readable media, such as a floppy disk, CDROM, memory card, FLASH memory device, RAM, ROM, or any other type of memory device or computer-readable media that interfaces to the device **200**. In another embodiment, the instructions may be downloaded into the device **200** from a network resource that interfaces to the device **200** via a data network. The instructions, when executed by the processing logic **208**, provide one or more embodiments of a system for automatically processing components on the device as described herein.

[0045] It should be noted that the method **300** illustrates just one embodiment and that changes, additions, or rearrangements of the method elements may be made without deviating from the scope of the invention.

[0046] **FIG. 4** shows transactions **400** that occur between a download server and a device during operation of one embodiment of a system to automatically process components on the device. For example, the transaction **400** may occur between the device **106** and the server **102** shown in **FIG. 1**.

[0047] At the start of the automatic process, the device **106** requests the latest version identifier of an action list from the server **102**, as shown at **402**. The action list comprises component/action pairs that describe an action the device should perform with respect to each identified component. The action list may be changed or updated

periodically and the action list version identifier identifies the current version of the action list.

[0048] The server **102** responds to the request from the device **102** by transmitting the version identifier of the current action list, as shown at **404**. After receiving the action list version identifier, the device compares that identifier with a stored action list version identifier. If the two version identifiers are equivalent, then the device takes no further action. If the two version identifiers are different, then the device **102** requests a new action list from the server **102**, as shown at **406**.

[0049] The server **102** responds to the request from the device **106** by transmitting the new action list, as shown at **408**. The device **106** processes each component/action pair in the action list to determine whether or not to install, update, or delete a particular component. If the device **106** determines that a particular component needs to be installed or updated, the device **106** requests the component (or update) from the server **102** as shown at **410**.

[0050] The server **102** responds to the request by transmitting the requested component to the device **106**. The device **106** receives the component and performs the installation or update as required. The component may have a component version identifier that the device stores locally. After the device **106** parses the entire action list and retrieves all the needed components from the server **102** as necessary, the device **106** updates a locally stored action list version identifier with the version of the action list that was just processed. Thus, the device **106** will not process another action list from the server **102** until the action list version identifier downloaded from the server is different from the stored identifier.

[0051] **FIG. 5** shows one embodiment of an action list **500** for use in a system to automatically process components on a device. The action list **500** comprises an action list version identifier **502** followed by component/action pairs (**504**, **506**). For example, component/action pair **504** comprises a component identifier and a corresponding action. In one embodiment, the component identifier also includes a component version identifier, so that the version of the component can be used to determine whether or not the component currently exists on the device. The action may be one of "install", "update", or "delete" however, any other type of action may be specified. The

information in the action list may be encoded using any suitable format and the component and action identifiers may be of any type.

[0052] Accordingly, while one or more embodiments of a system to automatically process components on a device have been illustrated and described herein, it will be appreciated that various changes can be made to the embodiments without departing from their spirit or essential characteristics. Therefore, the disclosures and descriptions herein are intended to be illustrative, but not limiting, of the scope of the invention, which is set forth in the following claims.

WE CLAIM:

CLAIMS

1. A method for automatically processing components on a device, the method comprising:

receiving a version identifier associated with an action list;

determining that the version identifier is not equivalent to a stored version identifier;

receiving the action list;

parsing the action list to obtain a component identifier and an associated action; and

performing the action on a component identified by the component identifier.

2. The method of claim 1, further comprising updating the stored identifier to be equivalent to the version identifier.

3. The method of claim 1, wherein the action is installing and the step of performing comprises installing the component on the device.

4. The method of claim 1, wherein the action is deleting and the step of performing comprises deleting the component from the device.

5. The method of claim 1, wherein the step of performing comprises changing the state of the component on the device.

6. The method of claim 1, wherein the step of parsing further comprises determining that a component version associated with the component identifier is not equivalent to a stored component version.

7. The method of claim 6, further comprising updating the stored component version to be equivalent to the component version.

8. The method of claim 1, further comprising receiving the version identifier, the action list, and the component from a server.

9. The method of claim 1, wherein the device is a wireless device.

10. Apparatus for automatically processing components on a device, the apparatus comprising:

logic to receive a version identifier associated with an action list;

logic to compare the version identifier to a stored version identifier;

logic to receive the action list if the version identifier is not equivalent to the stored version identifier;

logic to parse the action list to obtain a component identifier and an associated action; and

logic to perform the action on a component identified by the component identifier.

11. The apparatus of claim 10, further comprising logic to update the stored identifier to be equivalent to the version identifier.

12. The apparatus of claim 10, wherein the logic to perform the action comprises logic to install the component on the device.

13. The apparatus of claim 10, wherein the logic to perform the action comprises logic to delete the component from the device.

14. The apparatus of claim 10, wherein the logic to perform the action comprises logic to change the state of the component on the device.

15. The apparatus of claim 10, wherein the logic to parse that action list further comprises logic to determine that a component version associated with the component identifier is not equivalent to a stored component version.

16. The apparatus of claim 15, further comprising logic to update the stored component version to be equivalent to the component version.

17. The apparatus of claim 10, further comprising logic to receive the version identifier, the action list, and the component from a server.

18. The apparatus of claim 10, wherein the device is a wireless device.

19. Apparatus for automatically processing components on a device, the apparatus comprising:

means for receiving a version identifier associated with an action list;

means for determining that the version identifier is not equivalent to a stored version identifier;

means for receiving the action list;

means for parsing the action list to obtain a component identifier and an associated action; and

means for performing the action on a component identified by the component identifier.

20. The apparatus of claim 19, further comprising means for updating the stored identifier to be equivalent to the version identifier.

21. The apparatus of claim 19, wherein the means for performing comprises means for installing the component on the device.

22. The apparatus of claim 19, wherein the means for performing comprises means for deleting the component from the device.

23. The apparatus of claim 19, wherein the means for performing comprises means for changing the state of the component on the device.

24. The apparatus of claim 19, wherein the means for parsing further comprises means for determining that a component version associated with the component identifier is not equivalent to a stored component version.

25. The apparatus of claim 24, further comprising means for updating the stored component version to be equivalent to the component version.

26. The apparatus of claim 19, further comprising means for receiving the version identifier, the action list, and the component from a server.

27. The apparatus of claim 19, wherein the device is a wireless device.

28. A computer-readable media comprising instructions, which when executed by processing logic in a device, operate to automatically process components on the device, the computer-readable media comprising:

instructions for receiving a version identifier associated with an action list;

instructions for determining that the version identifier is not equivalent to a stored version identifier;

instructions for receiving the action list;

instructions for parsing the action list to obtain a component identifier and an associated action; and

instructions for performing the action on a component identified by the component identifier.

29. The computer readable media of claim 28, further comprising instructions for updating the stored identifier to be equivalent to the version identifier.

30. The computer readable media of claim 28, wherein the instructions for performing comprise instructions for installing the component on the device.

31. The computer readable media of claim 28, wherein the instructions for performing comprise instructions for deleting the component from the device.

32. The computer readable media of claim 28, wherein the instructions for performing comprise instructions for changing the state of the component on the device.

33. The computer readable media of claim 28, wherein the instructions for parsing further comprises instructions for determining that a component version associated with the component identifier is not equivalent to a stored component version.

34. The computer readable media of claim 33, further comprising instructions for updating the stored component version to be equivalent to the component version.

35. The computer readable media of claim 28, further comprising instructions for receiving the version identifier, the action list, and the component from a server.

36. The computer readable media of claim 28, wherein the device is a wireless device.