Technical Disclosure Commons

Defensive Publications Series

May 29, 2019

Transferring Application Software from One User Equipment (UE) to Another

Victor Carbune

Daniel Keysers

Thomas Deselaers

Transferring Application Software from One User Equipment (UE) to Another

Abstract:

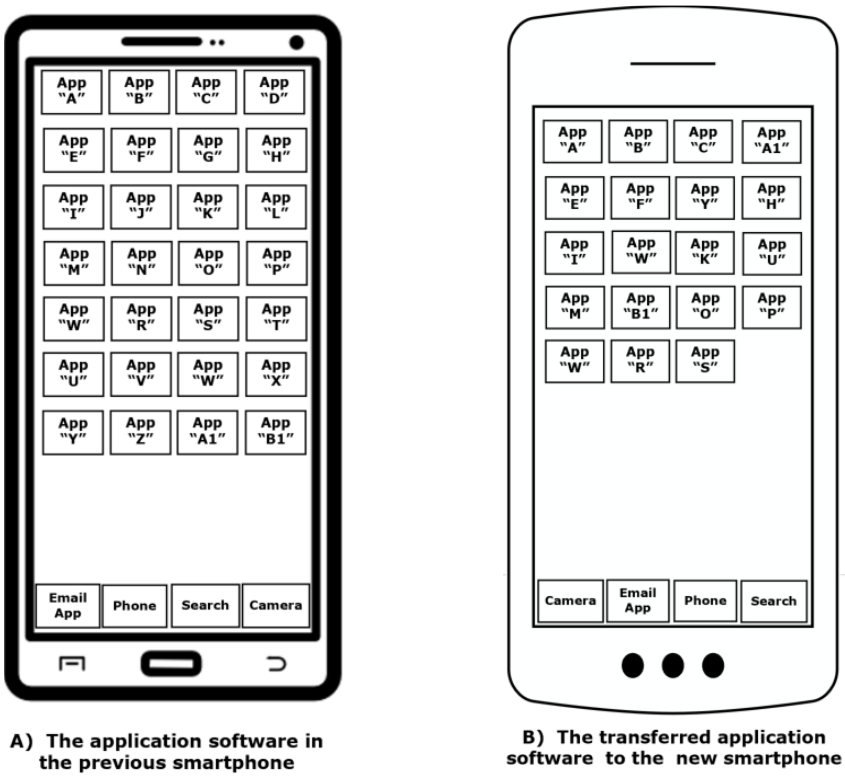
An operating system (OS) of a user equipment (UE), such as a smartphone, supports various first-party and third-party application software. Many application software offer personalized user experience (UX) depending on user settings. A user may spend a considerable amount of time and effort downloading the application software and configuring their settings. The user, however, periodically replaces the smartphone, and often they download and configure the settings of various application software again on the new UE, same as they had done on the previous UE. This leads to a lengthy process of migrating from one UE to another. This publication describes methods the operating system (OS) uses to aid the user transfer various application software and their settings from one user equipment (UE) to another. As described herein, transferring may refer to copying or moving. To transfer the user's application software and their settings from one UE to another, the OS enables application software developers to implement a system-level application programming interface (API). In the case when an application software developer does not implement or support the OS's system-level API, the OS uses a machine-learning model that aids the user to configure the settings of a non-initialized application software in the new UE, the same way they had done using the previous UE.

Keywords:

Application programming interface, API, user interface, UI, application software, user settings, customization, personalization, replicate, transfer, copy, move, duplicate, machine learning, supervised learning, reinforcement learning, artificial intelligence, AI.

Background:

An operating system (OS) of a user equipment (UE), such as a smartphone, supports various first-party and third-party application software. Many application software offer personalized user experience (UX) depending on user settings. The user may spend a considerable amount of time downloading the application software and configuring their settings. Often the user may settle on an application software's settings after they have used the application software for some period. The user, however, periodically replaces the smartphone for several reasons, such as to upgrade to a new smartphone model, to replace a broken, a lost, or a stolen smartphone, and so forth. When the user replaces the smartphone, they may want to transfer all application software and their settings from the previous smartphone to the new smartphone. Transferring the application software and their settings may be time-consuming, as illustrated in Figure 1.



A) The application software in the previous smartphone

B) The transferred application software to the new smartphone

Figure 1

Assume Tom gives Jane a new smartphone for Christmas. Jane is excited and wants to start using it at once. Jane unpacks the smartphone and goes through the setup flow of the smartphone. After Jane sets up the smartphone, she notices that not all the application software were transferred to the new smartphone, as seen in Figure 1. On the new smartphone, Jane downloads the missing application software she most-uses, and she starts configuring their settings as she had previously done. Up to this point, Jane has already spent a considerable amount of time and effort migrating to the new smartphone. In addition, since Jane does not use all the application software at once, she may notice the lack of setup (*e.g.*, missing username, missing profile photo, un-linked credit cards, default notifications settings) later. Jane may need to retrieve the previous smartphone and copy over the application software settings to the new smartphone. In the case when Jane may not have access to the previous smartphone because it may be broken, lost, or stolen, she needs to go through the application software setup process from the beginning.

Modern smartphones support copying some data from one smartphone to another, but the process is not smooth, and many application software may not properly support this process. This leads to a lengthy process of migrating from one UE to another. In addition, some application software may not allow the user to sign in to two UEs, such as two smartphones, or some application software may limit the number of UEs the user may sign in. It is desirable for the OS to aid the user to more-easily migrate from one UE to another.

Description:

This publication describes methods an operating system (OS) uses to aid a user transfer various first-party and third-party application software and their settings from one user equipment (UE), such as a smartphone, to another. The OS enables application software developers to implement a system-level application programming interface (API) that can transfer the user's application software and their settings from one UE to another. As described herein, transferring may refer to copying or moving.

As the user goes through the setup flow of the new UE, the OS's system-level API recognizes that the application software and their setting in the new UE do not match those in the previous UE. The system-level API alerts the individual application software that the user wants to transfer the application software and their settings to the new UE. The application software developers may utilize the OS's system-level API to transfer the user's settings in a structured way. In addition, the OS's system-level API gives the application software developers the ability to choose what data may be copied, moved, or prevented from copying or moving (e.g., temporary data, network cache data).

In certain cases, the application software developers may not allow the system-level API to copy the application software and their settings from one UE to another because they may offer services that may be restricted to one cellphone number (*e.g.*, voice over internet protocol (VoIP) calling or video-conferencing). Also, some application software developers may limit the use of their application to one (1) UE for security reasons. For example, the application software may handle highly-sensitive information, such as financial transactions, personal-sensitive information, or professional-sensitive information, and a business entity (*e.g.*, a law firm, an intelligence agency, a high-tech company) may purposely restrict the user to one (1) UE. In the case when the

use of an application software is restricted to one (1) UE, the OS's system-level API may offer the user the choice to move, instead of copy, the application software and their settings from the previous UE to the new UE. The OS's system-level API may help the user move data even when an application software places a condition on the user to switch a smart card inside (SIM) or an embedded-SIM card (eSIM) from one UE to another before the application software and their settings may be transferred.

The system-level API relies on the individual application software developers to adopt this solution. Both the OS developer and the application software developers have an incentive to enhance the UX with their products, including the case when the user starts using a new UE. Figure 2 helps illustrate the case when the application software developers implement the OS's system-level API to transfer data from one smartphone to another.

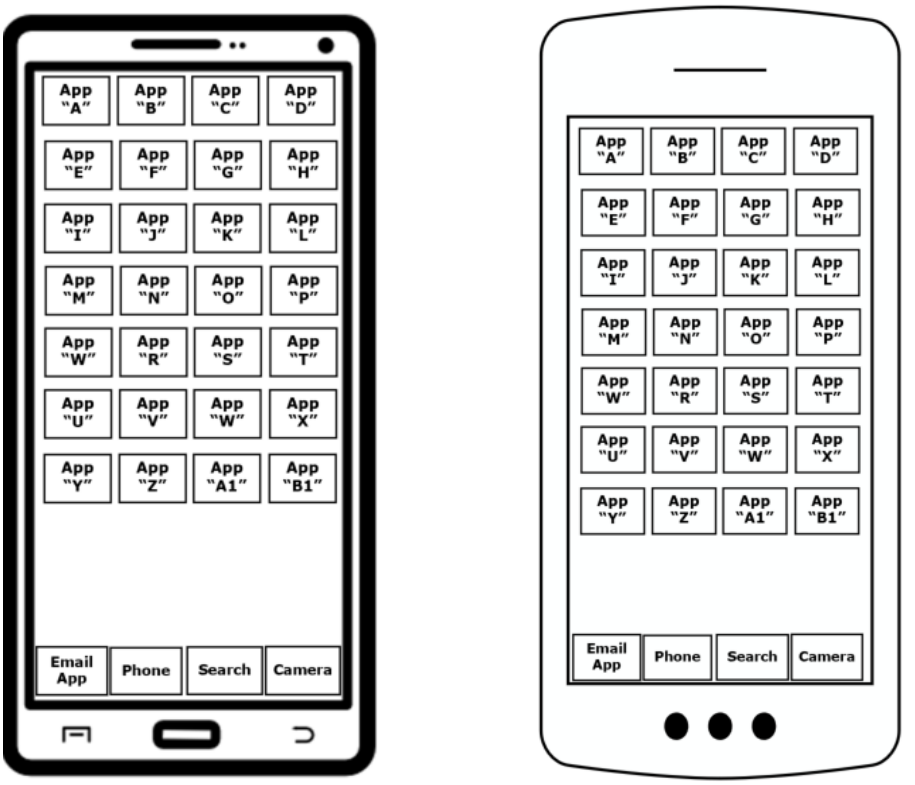


Figure 2

Unlike in the example of Figure 1, assume Jane is using a smartphone with the OS with the system-level API that helps her transfer the application software. As Jane unpacks the new smartphone and goes through the setup flow, the OS's system-level API transfers all application software to the new smartphone, as seen in Figure 2. In addition, all the application software settings in the new smartphone are the same as on the previous smartphone. Furthermore, Jane does not need access to the previous smartphone, which may be important in case Jane's previous smartphone was broken, stolen, or lost. Jane may start using the new smartphone without spending any extra time and effort and have the same UX with each application software.

In the case when an application software developer does not implement or support the OS's system-level API to transfer application software and their settings, the user can download the application software on the new UE. In the case when an application software developer does not permit the application to run on two different devices, the user may first need to log out or delete the application software from the previous UE. When the user is ready to configure the settings of the application software on the new UE, the OS uses a machine-learning model that aids the user to configure a non-initialized application software in the new UE, the same way they had done using the previous UE.

Figure 3 illustrates how the OS developer may use a plain recording of actions for simple interactions.

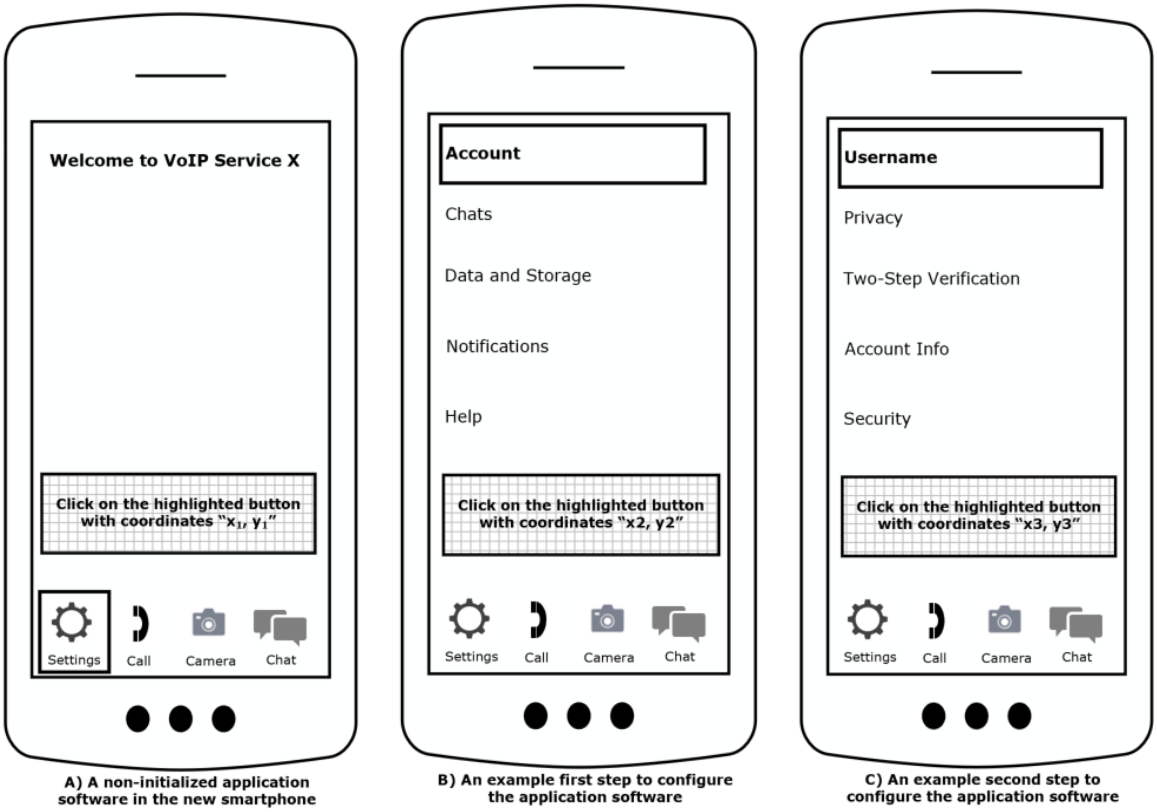


Figure 3

Assume Jane is using an application software that does not implement the OS's system-level API. As Jane unpacks the new smartphone and goes through the setup flow, the OS's system-level API does not transfer the application software and its settings to the new smartphone. Jane downloads the application software on the new UE and clicks on the icon of the application software. The application software does not recognize Jane as a current user and prompts her to configure it by saying, "Welcome to VoIP Service X," as seen in Figure 3A. The OS has recorded the actions Jane took to configure the settings of the application software on the previous smartphone. The OS, then, aids Jane by giving her instructions on how she previously configured the settings of the application software and using a toast or a pop-up says, "Click on the highlighted

button with coordinates "x_1, y_1," as seen in Figure 3A. After Jane clicks the button with coordinates x_1, y_1, the application software displays a different screen. Using another toast or pop-up, the OS instructs her on the second step of configuration by saying, "Click on the highlighted button with coordinates "x_2, y_2," as seen in Figure 3B. After Jane clicks the button with coordinates x_2, y_2, the application software displays yet another screen. Using yet another toast or pop-up, the OS instructs her on the third step of configuration by saying, "Click on the highlighted button with coordinates "x_3, y_3," as seen in Figure 3C. This process continues until Jane fully configures the settings of the application software as she had previously done.

Figure 3 is just one way the OS may aid the user to configure the settings of an application software on the new UE, but the OS may also use a standard-supervised machine-learning model for simple tasks, such as given a screen state, the user is asked to click one (1) button among "N" number of buttons. For more sophisticated interactions, the OS may use more-sophisticated machine-learning models, such as reinforcement learning, a neural network, a support vector machine, a recurrent neural network (RNN), a convolutional neural network (CNN), a dense neural network (DNN), heuristics, or a combination thereof. Given the large computational power that some more-sophisticated machine-learning models may use to train a model to analyze so many inputs, the model training may be performed on a cloud, server, or other capable computing device or system.

Further to the descriptions above, a user may be provided with controls allowing the user to make an election as to both if and when systems, programs, or features described herein may enable collection of user information (*e.g.*, a user's preferences or a user's current location), and if the user is sent content or communications from a server. In addition, certain data may be treated in one or more ways before it is stored or used, so that personally identifiable information is

removed. For example, a user's identity may be treated so that no personally identifiable information can be determined for the user, or a user's geographic location may be generalized where location information is obtained (such as to a city, ZIP code, or state level), so that a particular location of a user cannot be determined. Thus, the user may have control over what information is collected about the user, how that information is used, and what information is provided to the user. The user may also choose to ignore the described application software transfer methods and download and configure the settings of all application software without the help of the OS.

In conclusion, the implementation of the OS's system-level API to transfer application software and their settings from one UE to another helps the user shorten and simplify the process of migrating from one UE to another. In the case when an application software does not support or implement the OS's system-level API, the OS may use a machine-learning model to aid the user to configure the settings of a non-initialized application software in the new UE, the same way they had done using the previous UE.

References:

[1] Martin, Derek P. Transferring user settings from one device to another. US Pub. 20150067099, filed March 12, 2014, and published March 5, 2015.