Realtime publishers

The Essentials Series:
Delivering Pervasive User Profiles

Delivering Pervasive Personalization

sponsored by

RES software

by Greg Shields

Copyright Statement

Delivering Pervasive Personalization

The previous article in this series outlined the business benefits that can be gained through the implementation of a platform of pervasive personalization. As explained there, this platform enables the decoupling of user personality elements from individual OS instances. Through the implementation of such a platform, your business' IT organization stands to gain a number of workflow improvements. These improvements arrive through the automation and dissemination of user workspaces to wherever they are needed in the environment.

What this series hasn't yet done is to discuss exactly how this platform of pervasive personality might work from a technical perspective. How exactly could an organization implement technology that delivers pervasive personalization to users no matter how they interact with your IT environment? How would it be managed and how would it interact with the primary operating system (OS) that is installed to those user interfaces? This article will attempt to discuss how such a solution could be created.

In effect and at a very high level, three components are necessary to create a pervasive personalization platform:

- A database for the storage of individual personality elements.
- A transmission and synchronization component that manages the transfer of data between the database and the individual clients on the network.
- A mechanism to integrate the decoupled personalization elements with the installed OS.

These three elements provide a way to replace the existing solution for profiles that is provided natively in the Microsoft Windows operating system. By leveraging a wholesale replacement of its profile management and delivery subsystem, an entirely new set of rules can be created and managed for dealing with user personality. To gain a greater understanding of how this solution might be implemented, let's look at each of these elements in turn.

Requirement #1: A Database

The first article in this series discussed how the composition of a user's workspace is made up of various elements such as the desktop background, browser bookmarks, applications and application settings, printers, attached peripherals, and local files and folders. In the Windows OS, each of these elements is typically stored within the user's profile. Within this profile—stored locally in the subfolders of *C:\Users* on Windows Vista and Windows Server 2008 and in the subfolders of *C:\Documents and Settings* for earlier versions—are files and folders of data that relate to the user's customizations to the OS and installed applications. Also in this location is a file-based instance of the local Windows registry database for the user, stored in a file called *NTUSER.DAT*. The registry database includes additional user-specific settings for the OS and applications, all of which are stored in the registry hive HKEY_CURRENT_USER.

In environments that use only device-centric personalization, these individual instances are stored locally on each device with which the user interacts. Logging into a particular device loads the information from the profile into the current user's session as well as the user's registry. Logging into a separate device or interface loads the profile information that is stored in that location. Because in this case the profile information is unique to each interface, and there is no replication between interface instances, the user's workspace is unique as well.

Creating an environment of pervasive personalization first requires a separate database in which these configurations can be stored (see Figure 1). By housing the metadata associated with a user's profile in such a database, that information can be easily consolidated across and distributed to any number of login interfaces. The workspace associated with a user logon is generated by combining the structure of a local profile with the metadata in this remote database.

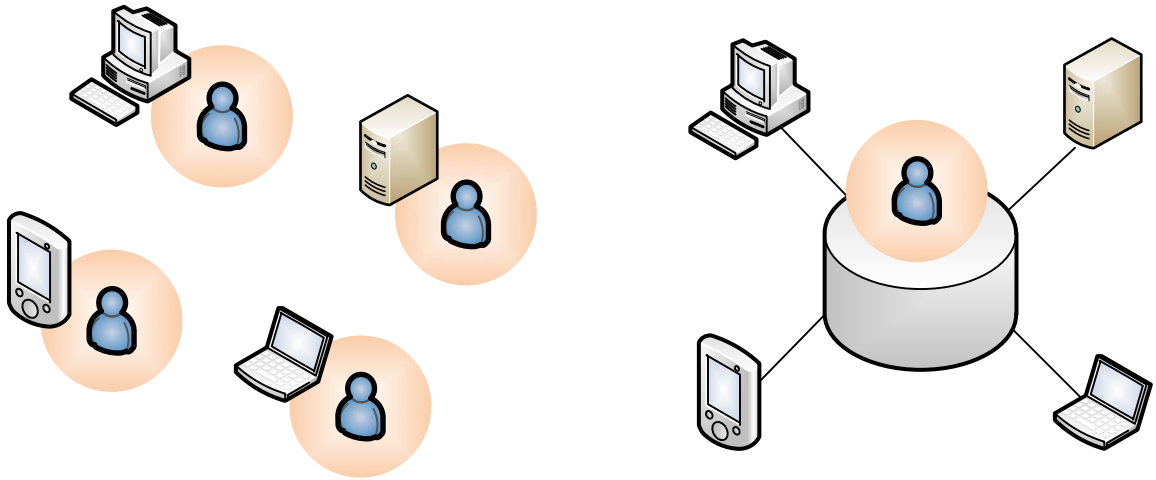

Figure 1: On the left is shown how traditional device-centric profiles replicate user workspace instances, while on the right a pervasive personality platform unifies instances with a central database.

In contrast, Microsoft Windows natively includes the capability to create roaming profiles, which at first blush appear to serve the same function. However, these profiles are quite different in operation. Windows roaming profiles are actually transferred in whole from a central storage location to the local device at the time of a user's login. Unlike in the database used in a pervasive personality platform, their central storage includes all the files, folders, and data that make up the user's profile. Thus, an entire profile must be transferred en masse over the network to the user's logon point. This brute-force approach can delay logons in the case of large profile sizes and can cause unexpected behaviors when users log onto and out of multiple machines at once. This occurs because the contents of each roaming profile data merged at the central storage point for each logon.

Another benefit of this decoupled database concept is in how its contents can be much more easily managed through standard administrative interfaces. Because such a system stores metadata associated with the user's workspace, it is possible for an administrator to pre-configure, lock down, or otherwise control certain aspects of the user's workspace. The IT organization can choose which

- Applications and their settings are brought down to the local machine
- Customizations are allowed and which are centrally controlled
- Interfaces have greater or lesser levels of control; for example, LAN versus Internet, day and time restrictions, local login versus remote application server, and so on

Requirement #2: Transmission and Synchronization

Because of the brute-force approach to user profiles in the native Microsoft Windows OS, there is no onboard mechanism for profile information to be rapidly transmitted to targeted user interfaces. Similarly, there is no mechanism to synchronize updates from one user interface to others. Thus, in order to accomplish this, an external client is required to be present on each OS instance that a user might log into.

This client is responsible for the handshaking and data transfer process that occurs during a user logon event. It communicates with the system's external database to download profile metadata as well as administrator policies and configurations. This occurs during the logon process and completes as the user gains controls of the session. This component can further ensure that configuration changes are synchronized between user interfaces through regular polling activity. This polling activity ensures that a change made in one interface eventually updates all other interfaces as well.

Because an external infrastructure is used for this activity, the performance of the logon process is greatly enhanced over what is seen with native tools. User profiles tend to be much smaller in size, which reduces the amount of time involved with the login process. The profile transfer process is also handled through a separate and isolated system, which makes user profiles significantly more secure during logons.

Requirement #3: OS Integration

The last components in a pervasive personalization platform are the necessary "hooks" into the native OS. These hooks make all this system's capabilities possible. Hooks into the system can be accomplished through a number of mechanisms. Profile metadata can be injected into local profile structures to ensure that the proper data endpoints are available for users to work with once they gain control of their login. For example, if an application setting is known to be needed in a certain registry location for that setting to impact the use of an application, the data associated with the setting can be injected at the time of logon into the proper place so that the registry is correctly updated during the logon process.

Managing the user workspace can be further customized through the use of an entirely separate mechanism for composing the user's workspace. Native tools such as Group Policy provide some level of control over the user's experience but are challenging to use and are not context-based. Needed are external solutions that run in parallel with user sessions and provide context-sensitive access to the right resources. This means that users are naturally connected to the printers closest to them; access the right data based on their identity, time of day, and location; and are connected to the right applications.

Implementing such a mechanism creates a platform for much more granular control over user workspaces. Personalization elements that comprise the user's workspace can be retained and stored as metadata in the external database as discussed earlier. Users that login to multiple interfaces can be assured that their experience has the same look and feel no matter where they login. Administrators retain a much greater ability to control necessary areas of the user's workspace as administrators see fit and their security or corporate policies specify.

Pervasive Personality Enables Pervasive Management

Smart organizations understand the need for a personalized workspace. At the same time, however, many may not immediately recognize that an optimized workspace is not only personalized but also secure and reliable. Organizations must see the need for an independent mechanism to handle personalization—one which is not possible through traditional systems management solutions alone. Only through the creation of a completely separate infrastructure for handling user profiles and their settings can organizations truly decouple their users' experience from their business infrastructure.