Network Configuration with Plug-and-Play Components

S. K. Raza
Systems & Computer Engineering
Carleton University, 1125 Colonel By Drive
Ottawa ON K1S 5B6, Canada
skraza@sce.carleton.ca

A. Bieszczad
Advanced Technologies, Bell Labs
101Crawfords Corner Road
Holmdel, NJ 07733, USA
bieszczad@lucent.com

The installation and configuration of a new network component is a difficult task due to the heterogeneous nature of today's telecommunication networks. First, a network administrator has to arrange for a hardware connection for the component, followed by the provisioning of software modules and adjustments of software attributes. Finally, the installed modules are to be set up and activated on the corresponding client components. All the aforementioned steps require substantial time, effort and extensive expertise from the network administrator, depending on the size, heterogeneity and other parameters of the network. One of the approaches to resolve the described problem is to ***automate*** the installation and configuration process of a new network component using the plug-and-play concept [1]. Therefore, we define plug-and-play (PnP) network components to be components that are capable of configuring both themselves and other cooperating network components without human intervention.

Our previous work [1] has suggested several ways of achieving plug-and-play capability using static agents. This paper takes the work onward but focuses on an implementation that uses mobile agent technology. We have used a Java based mobility infrastructure, called the 'Mobile Code Toolkit' [2], for building our application. Following is the suggested scenario that a typical network plug-and-play component (a printer in this example) would follow during its installation and configuration.

The vendor supplies a mobility-enabled network component that has a specific *plug-and-play module* installed on it. This interface is capable of initiating and controlling the auto installation process by sending and receiving the mobile agents. After manual installation and booting, the PnP module starts the configuration process by dispatching a mobile agent to the vendor site for the purpose of *registration*. After successful registration, another mobile agent is sent over the network to *discover* the configuration requirements of other network components. This discovery agent visits a network component, gathers the information related to its type and operating system, and then migrates to the next component using the default migration target. When the mobile agent returns to its origin, the plug-and-play component's software determines the configuration requirements of the

supported network and sends another mobile agent with these requirements (request for service modules) to the vendor site. The agent approaches a particular interface at the vendor site to access the vendor's repository for service modules, and then sends the requested modules back to the PnP component. After receiving the required service modules, the PnP component prepares and dispatches separate dedicated *setup* mobile agents towards each discovered component. These agents carry platform dependent service modules for the PnP component that are required by the client components. The modules are delivered at each component's setup interface that makes further arrangements for the installation (may store or discard) of the module. With this step, the initial cycle of installation and configuration of the PnP component is completed, and the other components are now ready to use the PnP component's service. There are two more requirements for the further support of the PnP component: a permanent discovery agent on the network that keeps roaming, looking for a change in the network configuration and topology, and provision of notifications from the vendor to the PnP component regarding software upgrades.

Having simulated the PnP approach in our Network Management Lab, we conclude that mobile agents represent a powerful computing paradigm that provides significant advantages over the static agent approaches. The mobile agent paradigm provides the *flexibility* and power that are essential requirements for our application due to its interactions with the dynamic environment of the network. The other significant advantages include dynamic configuration, auto discovery of network components, an asynchronous auto installation procedure, up to date service module provisioning due to upgrade notifications, and platform independence due to Java. There are, however, some drawbacks and issues related to the scheme. Security is the first and biggest issue that arises whenever a piece of code is to be sent across the network. This issue is addressed by our mobility framework [2] using standard security controls. There are other issues related to the overheads (requirement of running mobility framework and Java on each component), controlling the frequency and the density of permanent discovery agent, setting initial configuration parameters, and establishing migration targets. These issues are to be resolved and constitute our future work.

This research is supported by Communication Information and Technology Ontario (CITO), Canada.

References

[1]. A. Bieszczad, S. K. Raza, B. Pagurek, and T. White, *"Agent-based schemes for Plug-And-Play Network Components"*, in the proceeding of International Workshop on Agents in Telecommunications Applications (IATA'98), July 1998, Paris France, http://www.sce.carleton.ca/netmanage.

[2]. G. Susilo, A. Bieszczad, and B. Pagurek, *"Infrastructure for advanced Management based on Mobile Code"*, in the proceedings of the IEEE/IFIP Network Operations and Management Symposium NOMS'98, February 1998, New Orleans, Louisiana.