Implementing Remote Monitoring to the PeerHood Middleware

Arto Hämäläinen, Jani Wunsch, Jari Porras

Lappeenranta University of Technology, Lappeenranta, Finland

Introduction

Abstract- In this paper an approach to extend the mobile computing environment through the utilization of fixed devices is presented. Seamless use of fixed devices as a part of mobile computing environment helps to save the scarce resources, e.g. communication and computation capacity as well as battery lifetime, of the mobile devices. Our approach is based on the use of the middleware that enables proactive discovery of devices and services in the close vicinity of the mobile device. Proactive operation, such as the scanning of wireless networks, consumes considerable amount of resources. Our approach migrates these resource consuming tasks to the fixed device, which then informs the mobile device of the changes in the near vicinity.

Limited resources (computational resources, memory and battery lifetime) and unreliable connectivity are the two major constraints for the mobile computing. First of these constraints can be solved by using the existing static infrastructure as efficiently as possible. Mobile computing environment for a mobile device include typically also fixed devices. These fixed devices have better computational power, as well as mains current as their power source. Additionally, they have some kind of connection to wired network infrastructure.

We present an approach for the utilization of fixed resources in the mobile computing environment. Our approach is based on a middleware solution, PeerHood, taking care of the mobile's mobile computing environment and remote monitoring service, implementing migration of monitoring task to the remote resource.

PeerHood middleware

The PeerHood middleware is software for proactive management, i.e. discovery, monitoring and maintenance, of mobile computing environment. The main software components of PeerHood are the PeerHood daemon, library and network plug-ins. The PeerHood*daemon* component uses different network *plug-ins* to proactively search for remote devices and their services. Applications can utilize this service discovery, as well as connection management, through *PeerHood library* interface. The software architecture of PeerHood middleware is shown in Figure 1.

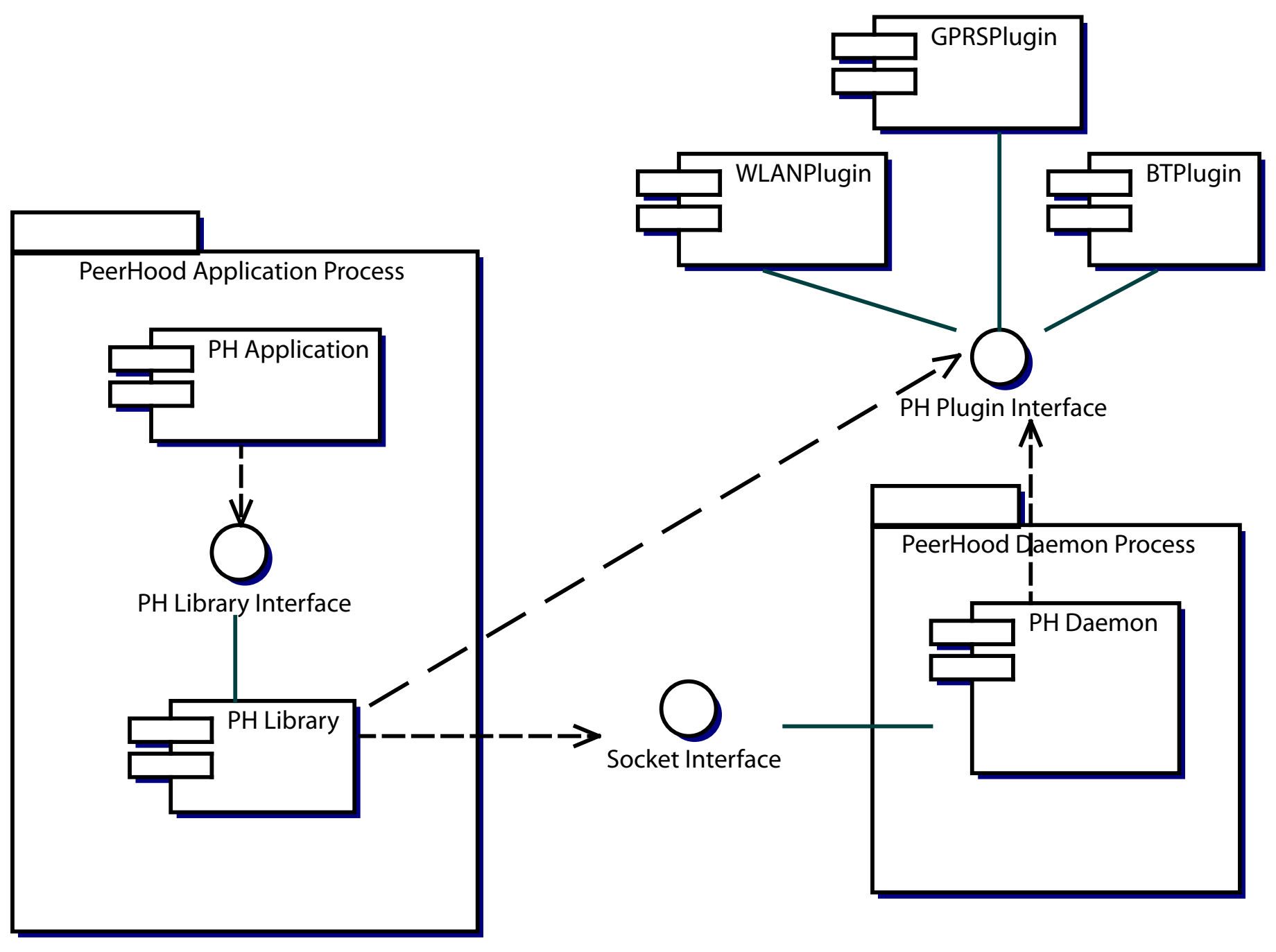

Figure 1: PeerHood software architecture

When the daemon process starts, it loads configured network plug-ins, that begin searching for neighboring devices and services, as well as responding to service requests sent by other devices. These plug-ins operate independently, but store their findings in the common device storage, maintained by the PeerHood daemon process.

Remote Monitoring

Remote monitoring in PeerHood is an approach to extend the mobile device operation through the utilization of fixed resources of the mobile computing environment. The idea is to use the fixed device to locate nearby services as long as the fixed device is available in the wireless neighborhood. The remote monitoring functionality is implemented as a PeerHood service, on top of the basic PeerHood operation.

Remote monitoring service informs mobile device of the changes in the environment through update procedure. Remote monitoring service in PeerHood provides two different levels of monitoring; *basic* and *advanced*.

- Basic operation of the remote monitoring takes care of the PeerHood device discovery process on behalf of other devices. Device discovery is performed with certain intervals, and service informs its clients to request updated list when available
- Advanced method of operation allows users to select number of devices for constant monitoring. Change in status of these devices is notified immediately

When mobile device wants to utilize remote monitoring service, it searches for the available service providers. PeerHood searches for the *rMon* service in neighbouring devices, and as soon as it finds the service, PeerHood requests user to select the desired service provider. Selected service provider is connected and mobile device may shut down its own operations. In this phase user needs to decide if advanced monitoring is used. The list of monitored devices needs to be sent to the remote monitoring service. If no devices are sent, rMon service reports only periodic device discovery results. The remote monitoring startup phase is presented in Figure 2.

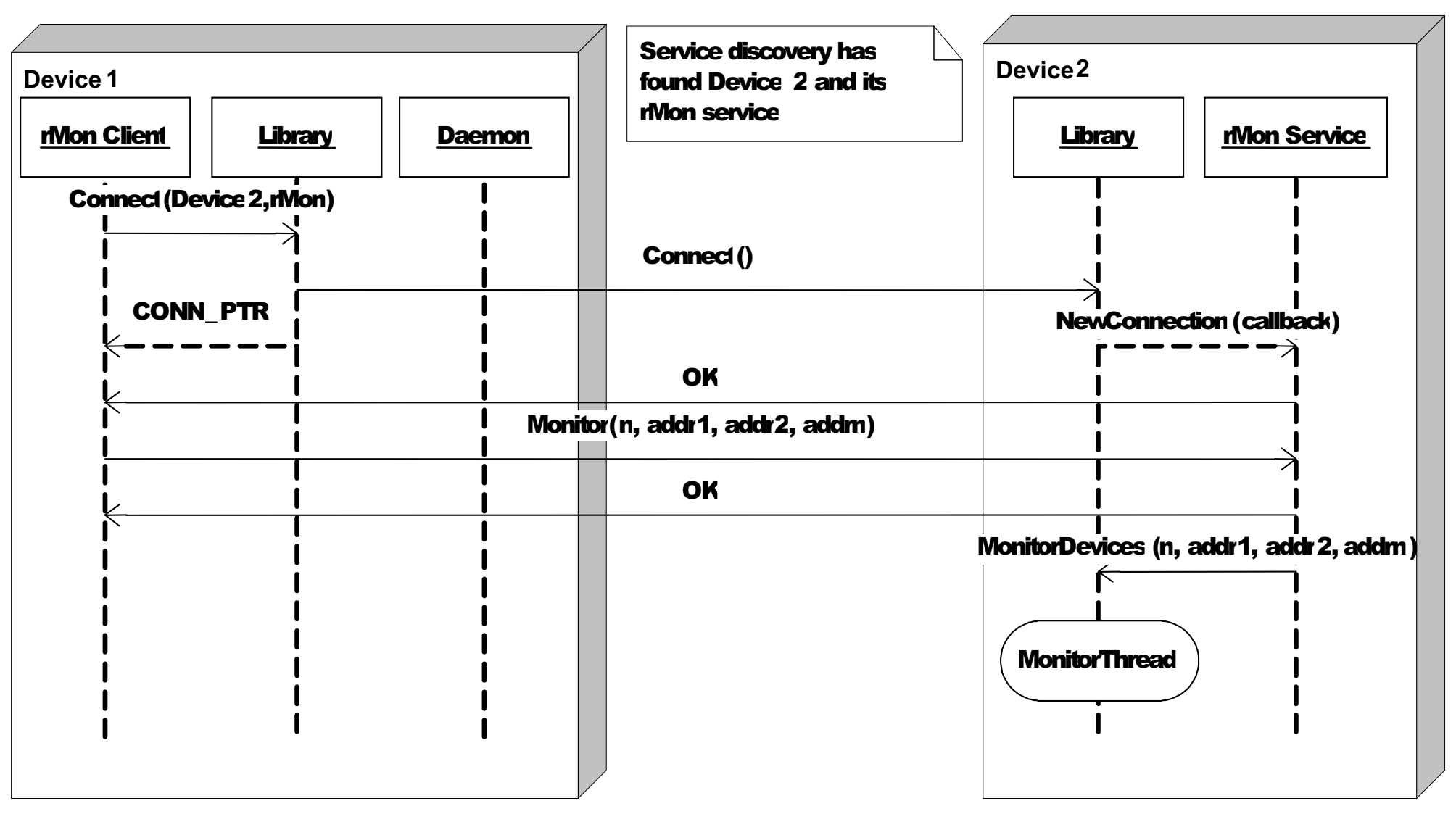

Figure 2: Starting remote monitor session

When mobile device is utilizing remote monitoring service, offered by a neighboring fixed device, the PeerHood daemon in mobile device instructs its network plug-ins to suspend the proactive device and service discovery. Even if mobile node has selected the advanced remote monitoring operation, i.e. to receive only status changes certain devices, it receives periodically acknowledgement messages from the service, confirming that the connection between devices still exists.

Updating device and service information using remote monitoring operation is pictured in Figure 3. The application, that has initiated the connection to the *rMon* service, starts listening for update messages from the *rMon* service. Device performing the monitoring task sends update message to client device to initiate device information request. Client repelies with a neighborhood information request, which will be answered by the remote monitoring service with a list of its known neighbors. This information is stored into the device storage of the client device.

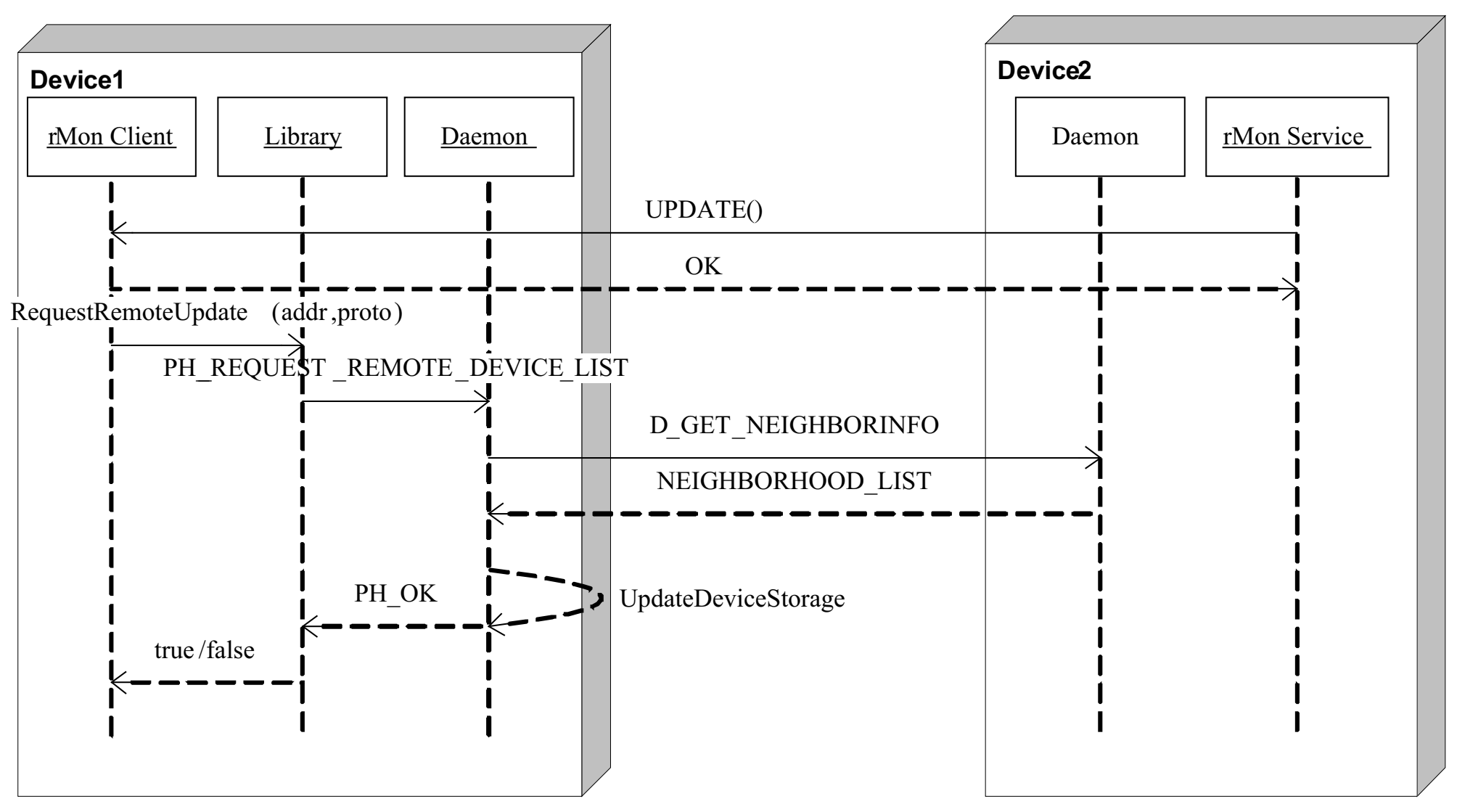

Figure 3: Updating client device with remote monitoring service

Remote monitoring is implemented as an optional service on top of PeerHood, and we want to keep it as optional. Therefore we wanted to keep the existing device discovery architecture with client initiated service discovery and left out the updated device list in the initial Update message. We keep the remote monitoring on the application/service level, instead of implementing it in the general functions of PeerHood. This has a cost of couple of messages to be exchanged, but no time-taking device discovery and connection establishment is needed, because the connection already exists between the devices. Advantage is, that no change is needed in PeerHood device discovery mechanisms implemented in network plug-ins.

Conclusion and Future work

Based on the preliminary results we have a strong feeling that the idea is highly usable and the implementation valid and efficient, especially, if the remote monitoring is tied together with some intelligent application considering the environment. Mobile devices are often located within certain fixed devices (e.g. work computer, home computer) during their daily operation. These devices could keep track of their changing neighborhood and offer information to mobile device located nearby.

Though our preliminary tests show that remote monitoring benefits the mobile device, it has to be remembered that remote monitoring is more of a tool than a final product. Fixed resources are able to monitor the changing mobile computing environment without the fear of running out of battery. With the update capability and direct notifications of applications PeerHood middleware enables environment-aware applications. The longer-term objective is to create an architecture with update and notification mechanisms to allow mobile devices truly use fixed devices in their mobile computing environment.

Lappeenranta University of Technology

Communications software laboratory

P.O.Box 20, FI-53851 Lappeenranta, Finland
http://www.lut.fi