Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 109C (2017) 319–326

Procedia Computer Science

www.elsevier.com/locate/procedia

The 8th International Conference on Ambient Systems, Networks and Technologies (ANT 2017)

Evaluating a Dynamic Keep-Alive Messaging Strategy for Mobile Pervasive Systems

Björn A. Johnsson*, Mattias Nordahl, Boris Magnusson

Department of Computer Science, Lund University, Ole Römers väg 3, SE-223 63 Lund, Sweden

Abstract

Timely loss-of-contact detection between devices in pervasive systems using mobile networks is an important aspect for both functionality and user experience. Traditional schemes where time-to-live is defined by the service provider are not adequate in mobile networks where communication failures are frequent. On the other hand, schemes using keep-alive messaging tend to increase the communication overhead, in particular if the time to detect a loss of contact needs to be short. We introduce a strategy where the time between keep-alive messages is adjusted dynamically according to the need in a particular application. Our solution was evaluated with usage data from a professional application built on the PalCom framework. The measurements show a 96 % decrease of the communication overhead while still maintaining the same system responsiveness.

Peer-review under responsibility of the Conference Program Chairs.

Keywords: Keep-alive messaging, loss-of-contact detection, heartbeat, mobile, pervasive systems, e-health, PalCom, middleware

1. Introduction

In stationary computer networks, service discovery has in many cases replaced manual configuration, e.g. entering network addresses. A common example is finding a printer on the local network. In such situations it is uncommon that the discovered device suddenly disappears before we actually come to use it. Hence, the nuisance when it does happen can be tolerated. In mobile settings it is, however, a common situation that connectivity to devices come and go. In many situations it is important for the user to be aware of whether a device and its services are available or not. The problem is thus not only to detect when a device become available, i.e. *discovery*, but also when a device become unavailable. We refer to the latter as *undiscovery*. In systems where mobile devices might become temporarily unavailable, these discovery and undiscovery mechanisms need to work together in order not to inflict penalties in terms of rollbacks when there are short breaks in communication. The detection of devices becoming unavailable affects not only the functionality of pervasive systems, but also allows for the user to be made aware of when functionality is available and when it is not, and why, through a user interface.

* Corresponding author. Tel.: +46-46-222-9642.
E-mail address: Bjorn_A.Johnsson@cs.lth.se

Peer-review under responsibility of the Conference Program Chairs.
10.1016/j.procs.2017.05.358

In this paper we build upon the mechanism that is used for discovery and undiscovery in the PalCom framework[1]. PalCom provides middleware support, and has been used for building a number of applications in e-health. We present a few of these, where the leading one is used to support nurses in home care with tablets for communicating patient information back and forth to a server, and for communication between the medical staff. One of the problems with the discovery mechanism is that it inflicts a communication overhead in order to detect loss-of-contact in a timely manner; there is a conflict between system responsiveness and bandwidth usage. In this paper, we introduce a strategy for detecting loss-of-contact, where the discovery parameters are adjusted dynamically to meet the needs of particular situations. From one of the e-health applications where PalCom has been used, we present measurements of actual application usage. We use these to estimate how much communication overhead can be saved in that specific case, with the purpose of illustrating the general efficiency of the presented dynamic strategy.

2. Background

In pervasive computing, several middleware frameworks have been proposed by researchers and developers. The problem space is vast and no single middleware provide solutions in all areas. Typically some form of discovery protocol is provided for automatically discovering other devices or services, and there are many variations developed for different contexts and uses. One branch of discovery protocols are based on broadcasting service information, used e.g. by Allia, GSD, DEAPspace, DSD and Konark[2]. Some middleware limit their broadcast range or perform selective forwarding in order to utilize the network bandwidth more efficiently, whereas others broadcast their service descriptions and meta-data to all known devices. Service descriptions are usually cached and marked with a time to live, after which the information needs to be communicated again. The latter approach may be suitable for small and highly mobile networks, but also has the drawback of flooding the network, and thus scales poorly into larger networks[3].

In addition to discovery, there is also a problem of detecting device and network failures, or undiscovery, i.e. that a service or other resource is no longer available. One solution is to send periodic heartbeats[4], or keep-alive messages, to which the remote side replies with an acknowledgment, thereby verifying that their link still works. If no heartbeats or heartbeat replies are received for some specified amount of time, the link to the remote resource can be considered gone and some action needs to be taken, often removing the node from the set of known devices on the network. Several pervasive middleware uses this functionality, or some variation of it, to provide fault detection[5,6,7,8]. Other examples include TCP, where implementations may offer a keep-alive setting[9], and an extension to TLS and DTLS from 2012 introduced heartbeats as a way to avoid costly renegotiations[10].

2.1. The PalCom Discovery Mechanism

In PalCom, heartbeats serve to enable both discovery and undiscovery[1]. Upon booting, or connecting to a new network, a PalCom device broadcasts a heartbeat message to every other device on the network. Each receiver of such an initial heartbeat message broadcasts a heartbeat of its own. In a network with n devices, such a round of heartbeats will produce n broadcasted messages to make all devices aware of each other, assuming no dropped messages. A connected device is discovered by the other devices as soon as communication delays permit.

Devices may continue to periodically broadcast heartbeats, to which other devices reply, thus providing fault detection in case of e.g. a device crashing or loosing its network connection. Devices that only offer services, rather than depend on services offered by others, would typically not send heartbeats spontaneously, but only reply to those it receives. On the other hand, devices that do depend on services on remote devices, will send heartbeats to ensure that the devices it relies on are still available. They do so frequent enough to report undiscovery in a timely manner; an internal clock determines when to send heartbeats. If there are multiple devices that send heartbeats on the same network, only one needs to initiate a new round of heartbeats since all other devices will send a reply instead, and reset their clock. The heartbeat frequency for the whole network is therefore determined by the device with the hardest time constraints. In a mobile network, where messages are frequently dropped, this scheme ensures that devices will be discovered and short loss-of-contact will be repaired in the next round of heartbeats.

The heartbeat frequency is set for each PalCom device, and determines how long to wait before initiating a new round of heartbeats. Adjusting it allows users to achieve a balance between bandwidth usage and detection delay