Daniel Burgstahler, Ulrich Lampe, Nils Richerzhagen and Ralf Steinmetz: *Push vs. Pull: An Energy Perspective.* In: Proceedings of the 2013 6th IEEE International Conference on Service Oriented Computing & Applications (SOCA 2013), p. 1-4, Institute of Electrical and Electronics Engineers (IEEE), December 2013. ISBN 978-1-4799-2701-2.

Push vs. Pull: An Energy Perspective

Daniel Burgstahler, Ulrich Lampe, Nils Richerzhagen and Ralf Steinmetz
Multimedia Communications Lab (KOM), Technische Universität Darmstadt, Darmstadt, Germany
Email: {firstName.lastName}@KOM.tu-darmstadt.de

***Abstract*—In many application scenarios, such as traffic guidance or ambient living, services need to notify mobile applications about status changes. Such notifications to mobile devices can be realized using two principal approaches, namely *push*- and *pull*-based. Apart from functional differences, the two options likely result in different energy consumption, which is an important aspect due to the battery constraints of contemporary mobile devices. This paper provides a detailed assessment of energy consumption in pull- and push-based notification scenarios, considering different payload sizes and notification intervals. Our results indicate that an educated choice among both options may, depending on the specific application scenario, facilitate energy savings of up to 19%.**

I. Introduction

Today, mobile devices have developed into ubiquitous devices in everyone's pocket. For example, the number of smarthpones that are used around the world is continuously increasing and will reach 1.4 billion devices by the end of this year [1]. The increasing popularity and the ever-growing amount of built-in functions leads to continuously upcoming new functionality and services. The concept of easy-to-install apps allows rapid integration of new services. The integration of external information, e. g., about the environment and surroundings, allows new concepts for location-aware services.

Contemporaneously, the deployment of sensors in cities and buildings enriches the environment with a kind of intelligence. Cities start to become smart with respect to information and communication technology infrastructure and become able to provide near real-time information about the status of city infrastructures with high granularity. Traffic light information, inductive loop traffic detectors, as well as sensors observing parking spaces up to a level of single parking lots. The deployment of such systems is as well pushed by research projects, e. g., in the city of Santander, Spain, the deployment of 12,000 sensing devices has already started [2], [3]. Such systems form the foundation to provide environmental information and thus, are the enabler for value-added mobility services and applications. With respect to end users, an important application scenario is the notification about changes in traffic situations, congestion warnings, speed recommendations to use green waves, and dynamic routing to the next free parking lot. How information infrastructures must be designed to inform road users about relevant information of their environment is currently a focus in research projects [4].

Many upcoming application scenarios have one thing in common: The notification of users about new information cause the need of wireless communication, which has a relatively high energy demand. An important aspect of mobile devices is the dependency on batteries. Until today, the computing power of mobile devices has increased corresponding to Moore's law, which says that computing power doubles roughly every two years. However, the battery development hangs strongly behind and battery capacity "did not even double over the last decade" [5]. Already at the beginning of the rise of mobile devices it was obvious that the integration of so much functionality into one device must result in a high energy consumption and thus, shorten battery lifetime [6]. Especially all wireless communications have to be optimized because of the high energy demand. Different paradigms exist to implement previously mentioned notifications in mobile clients, namely *push*- and *pull*- based notification, that have different energy characteristics. Hence, the research question we aim to empirically answer in this work is: "What is the difference in energy consumption between pull- and push-based client notification?"

In the next Section II, we introduce the experimental setup, followed by the presentation and discussion of our results in Section III. In Section IV we give an overview of related work, and finally conclude our paper with a summary and outlook in Section V.

II. Experimental Design

In this section, we describe the overall design of our experiment. To begin with, we briefly explain the dependent and independent variables that were considered. Subsequently, we introduce our measurement tool and briefly describe its technical implementation.

A. Considered Variables

As described in the previous section, in this work we focus on the energy demand of client notifications . Thus, the energy consumption constitutes the only *dependent variable*. More specifically, we consider the average power consumption, which is the consumed energy divided by the execution time.

The dependent variable in our experiments, energy consumption, may potentially be determined by various factors, i. e., a set of *independent variables*. With respect to the main subject of our research, i. e., the notification paradigm, we distinguish between the following two options: First, in a *pull*-based invocation pattern for client notifications, new information, received from an external source or service, is buffered at a central server. The client periodically establishes a connection to the server to ask for information updates (cf. Figure 1). Second, in a *push*-based invocation pattern, the client first establishes a connection to the central server. A common technique for this is to use a modified TCP connection and open a socket with longer timeouts. This connection is kept open all time, independent from notification messages. To prevent the network devices in between to close

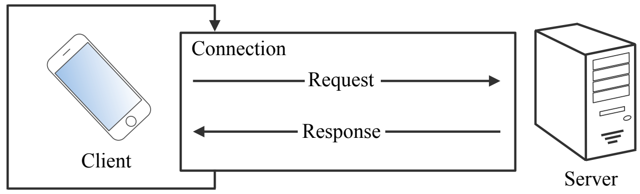

Figure 1. Schematic representation of *pull* notification of mobile clients.

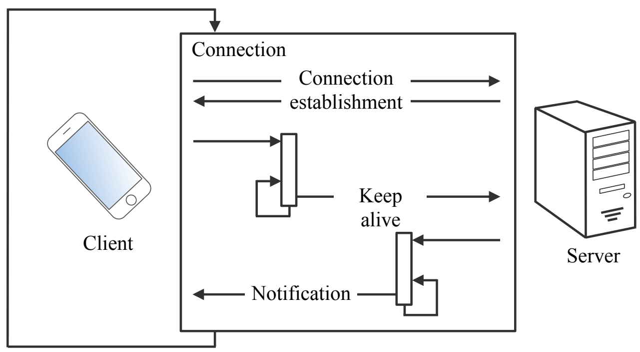

Figure 2. Schematic representation of *push* notifications of mobile clients.

the open connection (e. g., routers drop connection entries after a timeout if no new packets are transmitted), the client must regularly send keep-alive messages to the server. Once the server obtains updated information for the client, this can be transmitted immediately (cf. Figure 2).

With respect to the payload size, we distinguish between the three options: First, *small*, with a payload size of 10 Bytes, second, *medium*, with a payload size of 100 Bytes, and third, *large*, with a size of 1,000 Bytes. Concerning the update interval, we distinguish between the following two options: First, a *small* interval of 180 s, which corresponds to the minimum permissible interval in Google Cloud Messaging (GCM). Second, we chose a *large* interval of 1200 s. We selected the latter interval because it is much longer than the TCP connection timeout, but still provides a frequent client notification.

As platform for the implementation of our experimental application, we used Google Android. For the implementation of *push* notifications, we employed GCM[1]. This is a *push* service provided by Google, which is deeply integrated into the mobile operation system to minimize the energy consumption. Using GCM, the connection is maintained by the mobile operation system and a Google server forwards the notifications, which it receives from the initial notification server, to the client. Similar implementations of notifications services exist for all other commonly used mobile operating systems, such as Apple iOS[2] or Windows Phone[3]. For the implementation of *pull* notifications, we used our own server, which provides an Apache Tomcat[4] Web server with a RESTful interface that supplies information in JSON data format. Since GCM uses Transport Layer Security (TLS) to encrypt the communication between the mobile device and the GCM server, we have also encrypted the *pull* requests and responses using TLS.

B. Measurement System

Measuring the energy consumption of mobile device tasks can be done in several ways. The simplest way is to use a software tool. There are several applications available in the respective application market stores. These applications measure the execution time of the different mobile device components and estimate the energy demand based on an energy model. There also exist more sophisticated software solutions like PowerTutor [7] that are able to automatically determine a device specific energy model, but all these software solutions are only suitable for estimations, not for precise measurements. Thus, to get a precise measurement, one has to use an accurate external measurement device that is able to sample with a high frequency, since energy consumption states can change fast, e. g., when short messages are transmitted.

Additionally, one has to consider that modern mobile devices communicate with their batteries. This leads to two effects; on the one hand, the internal energy measurement can depend on the battery status, on the other hand, a mobile device cannot simply be powered by an external power supply, since it will not start up because of the missing feedback from the battery. Hence, to get measurements under realistic conditions, one has to power the device by the original battery.

We produced a dummy battery which exhibits the contacts and connected the electrical contacts of the dummy to the ones of the original battery with wires, a similar setup as proposed in [8]. To connect the wires to the original battery, we used a modified charging cradle. The dummy battery was placed in the phone to connect the electrical contacts to the phone. Now, we were able to break the circuit between the original battery and the dummy battery in the mobile device to place our measurement device in between. A schematic representation of the electric circuit is shown in Figure 3.

As measuring device, we used a Hitex Power Scale[5] with an *Active Current Measurement* (ACM) probe. The ACM system does not need any additional measuring resistors, whose usage would distort the result. The Power Scale measurement device has a high accuracy (up to nano-amps) and a maximum sampling rate of 100 kHz. For our measurements, we used a sampling rate of 10 kHz, which was previously found to be sufficiently high. The used Device Under Test (DUT) during our measurements was a Samsung Galaxy S III i9300 mobile device operating under Android 4.1.2.

C. Measurement Procedure

For our experiment, we followed a so-called *fractional factorial design* [10]. That is, we conducted measurements for a limited set of combinations of values for the three independent variables. We measured both notification paradigms with all three payload sizes with a constant notification interval of 180 s. Additionally, we measured both notification paradigms with a constant payload of 100 Bytes and a constant notification interval of 1200 s. Thus, our experimental setup consists of eight different *runs*. We sampled the power consumption with a

[1] http://developer.android.com/google/gcm/

[2] https://developer.apple.com/library/ios/documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/Chapters/ApplePushService.html

[3] http://msdn.microsoft.com/en-us/library/hh221549.aspx

[4] http://tomcat.apache.org

[5] http://www.hitex.com/index.php?id=powerscale

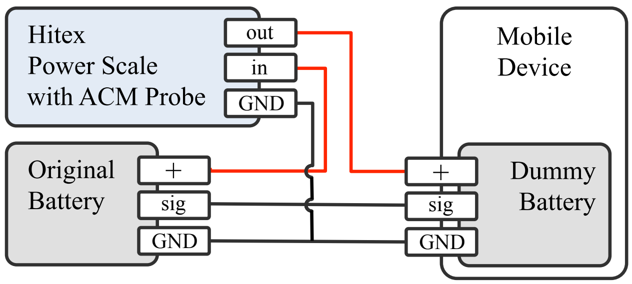

Figure 3. Electric circuit of the measurement system, with ground (GND), voltage (+), and signal (sig) lines. The latter indicates, e. g., the battery size and the current temperature of the battery via a Negative Temperature Coefficient Thermistor (NTC).

Table I. COMPARISON OF THE MEAN POWER CONSUMPTION FOR ALL CONSIDERED RUNS (SAMPLE SIZE $n = 5$). P-VALUES ARE BASED ON A MANN-WHITNEY U TEST [9].

Paradigm	Run Payload Size	Interval	Avg. Power [mW]	p-Value
Pull	10 Bytes	180 s	91.5	0.0556
Push	10 Bytes	180 s	90.2	
Pull	100 Bytes	180 s	90.2	0.0079
Push	100 Bytes	180 s	87.8	
Pull	1000 Bytes	180 s	90.4	0.2222
Push	1000 Bytes	180 s	90.9	
Pull	100 Bytes	1200 s	91.4	0.0079
Push	100 Bytes	1200 s	112.8	

sampling rate of 10 kHz; each run was repeated five times with a duration of 90 minutes, leading to 54 million observations per repetition.

As data connection, we used WiFi. To reduce external interference, we employed a separate access point and placed it in close proximity to the mobile device. There were no other wireless devices within the same room and we set the wireless channel of the access point to a number that was unused within the surrounding. We started our measurements a few minutes after initiating the experimental notification application to wait until the device went into power save mode and only our application was running in the background. Received notifications were only written to a log file; all vibration, optical, or acoustical notifications were disabled to prevent any influence on the measurements.

III. EXPERIMENTAL RESULTS AND DISCUSSION

In our first consideration, we focused on the influence of the payload size, which we raised from 10 Bytes to 1000 Bytes, while maintaining a constant notification interval of 180 s. The experiments were repeated for both notification paradigms. The results are given in the upper part of Table I. For all combinations, the average power consumption is about 90 mW. It can clearly be seen that the energy consumption for transferring data is independent of the used notification paradigm, with only a variation of about 1-3% depending on the used paradigms and payload sizes that can be considered noise. This also applies to the two runs with a payload size of 100 Bytes, where the difference between *push* and *pull* is statistically significant at $\alpha = 0.05$ according to a Mann Whitney U test, but where the absolute difference is still negligible.

However, in the case of a larger message update interval, i. e., 1200 s, the energy consumption strongly differs depending on the notification paradigm. It appears that for the *push*-based mechanism, the overhead of maintaining the connection to the server substantially contributes to the total amount of energy consumption, hence giving the *pull*-based mechanism – which exhibits practically static energy consumption independent of the payload size and update interval – advantages. Specifically, the lower part of Table I shows that for the larger update interval, the average power consumption of the *pull* mechanism is about 19% lower than for *push* mechanism. The result is statistically significant at $\alpha = 0.05$, according to the applied U test.

In conclusion, our results show that for a fixed amount of notifications and static notification intervals, the *pull* approach may exhibit a much lower energy demand. A further advantage of the *pull*-based approach is that it better suits small update intervals, since the communication is directly conducted between the application server and the mobile device, with no external server in between. However, using a *push* mechanism provides the clear advantage of no delay in message delivery. If a new information arrives at the messaging server, the message is directly transmitted since the communication channel is kept open at all times. Which mechanism is best suitable strongly depends on the application scenario. In case of exclusive externally determined scenarios, e.g, messaging services like WhatsApp[6], the *push* mechanism has clear advantages. Using a *pull* mechanism in such a scenario would lead to large delays in message delivery or a large number of continuous client requests, which will cause an increase of the energy consumption. However, in case of an application scenario where the client can determine a context that indicates the need of information updates, e. g., the search for a free parking lot, the *pull* mechanism can have advantages with respect to energy efficiency.

We will illustrate the difference with the following example, which is inspired by our ongoing work in the European Union-sponsored SIMPLI-CITY project [4]: A user starts a journey to a city and needs a free parking lot next to the desired destination. If a *push* mechanism would be used in such a scenario, the mobile device would register for information updates of the target area at the beginning of the journey, maintain the connection open, and will receive the according information. If a *pull* mechanism would be used in such a scenario, the mobile device can start a request every time the user is next to a parking lot to navigate the user as closely to the desired destination as possible. Here, the mobile device can determine relatively exactly the point in time when an information update is needed. In this case, the update interval plays a major role since driving with 50 km/h can result in a distance of up to 2.5 km in three minutes. Since in this example, the mobile device can determine the need of an information update depending on the current position, the *pull* mechanism has the potential to reduce the energy demand of the mobile device.

A further advantage with respect to energy savings could be a selective encryption. Whereas operation system built-in *push* mechanisms encrypt all communication, a mobile device could

[6] http://www.whatsapp.com

easily switch the encryption mode depending on the context when using the *pull* mechanism, since the message exchange is initiated by the mobile device.

IV. Related Work

Although energy consumption is a major concern for mobile devices, the difference in energy consumption of *push*- and *pull*-based notification mechanisms has not been extensively evaluated. Zhao et al. [11] investigate how push notifications can be used for command dissemination in mobile botnets; they also investigate the heartbeat traffic to maintain the connection between the mobile client and the notification server. Balasubramanian et al. [12] have shown the linear correlation between the amount of data and energy consumption for wireless transmissions, showing that the higher the amount of data, the higher is the caused energy demand. The influence of 802.11 network beacons on energy consumption is analyzed by Krashinsky et al. [13] which is very similar to keep alive messages used to maintain the connection in *push* implementations. They show the influence of beacon frames to the sleep interval of the devices and the respective increase of energy consumption in case of short intervals. A similar result is shown by Sharma et al. [14]. The impact of the different entities of a mobile device on the energy consumption is analyzed by Perrucci et al. [5]. They compare, amongst other things, the energy demand of communication components in different states. Hasenfratz et al. have shown that *pull* can save energy compared to *push* in data collection scenarios [15].

In summary, to the best of our knowledge, our work is the first to empirically analyze the difference of *push* and *pull* with respect to energy consumption of mobile devices. Our results enable us to determine the appropriate notification paradigm with respect to energy efficiency depending on the application scenario.

V. Summary and Outlook

The ubiquitous availability of high-speed data connections has tremendously changed functionality of mobile devices. Users can receive information updates in near real-time and updated information can easily be used to modify journeys and guide people intelligently through cities. To realize such functionality, information updates have to be transmitted to the user, which influence the energy consumption of the user's mobile devices. To reduce battery drain, developers have to decide about an appropriate notification paradigm.

In this work, our focus was on the experimental evaluation of energy consumption of mobile device notifications depending on different notification paradigms, namely *push* and *pull*. We have shown that the energy demand of a notification paradigm is independent of the payload size, but strongly influences energy consumption during longer periods without notifications. Depending on the application scenario, maintaining an open connection can cause more energy drain than transferring data at specific points in time. Even in our static example of a notification interval of 1200 s, we were able to show a difference in average power consumption of 19% between both notification paradigms. The energy consumption of a notification paradigm strongly depends on the amount of messages, temporal distance, and the possibility of the client to determine a context that indicates the need of information updates.

In our future work, we aim to substantially extend our experiments through the consideration of measurements of longer intervals of whole days with a variation of the distance between notification intervals, including series of short rapid notification sequences. We additionally strive to analyze the effects of different connection technologies such as GSM, 3G, and LTE.

Acknowledgements

Parts of the research leading to these results have received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 318201.

References

[1] Heather Leonard, "1.5 Billion Smartphones In The World - Business Insider," Feb 2013, online at http://www.businessinsider.com/15-billion-smartphones-in-the-world-22013-2.

[2] Libelium, "Smart City Project in Santander Monitors Environmental Parameters and Parking Slots," Feb 2013, online at http://www.libelium.com/smart_santander_smart_parking/.

[3] L. Sanchez, J. Galache, V. Gutierrez, J. Hernandez, J. Bernat, A. Gluhak, and T. Garcia, "SmartSantander: The Meeting Point Between Future Internet Research and Experimentation and the Smart Cities," in *Future Network Mobile Summit*, 2011.

[4] SIMPLI-CITY Consortium, "SIMPLI-CITY The Road User Information System of the Future," Oct 2013, online at http://simpli-city.eu.

[5] G. P. Perrucci, F. Fitzek, and J. Widmer, "Survey on Energy Consumption Entities on the Smartphone Platform," in *2011 IEEE 73rd Vehicular Technology Conf.*, 2011.

[6] R. N. Mayo and P. Ranganathan, "Energy Consumption in Mobile Devices: Why Future Systems Need Requirements Aware Energy Scale-Down," in *3rd Int'l. Conf. on Power-Aware Computer Systems*, 2004.

[7] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. P. Dick, Z. M. Mao, and L. Yang, "Accurate Online Power Estimation and Automatic Battery Behavior Based Power Model Generation for Smartphones," in *8th IEEE/ACM/IFIP Int'l. Conf. on Hardware/Software Codesign and System Synthesis*, 2010.

[8] A. Rice and S. Hay, "Measuring Mobile Phone Energy Consumption for 802.11 Wireless Networking," *Pervasive Mobile Computing*, vol. 6, pp. 593–606, 2010.

[9] R. E. Kirk, *Statistics – An Introduction*, 5th ed. Thomson Wadsworth, 2008.

[10] G. E. P. Box, J. S. Hunter, and W. G. Hunter, *Statistics for Experimenters. Design, Innovation, and Discovery*, 2nd ed. Wiley, 2005.

[11] S. Zhao, P. P. C. Lee, J. C. S. Lui, X. Guan, X. Ma, and J. Tao, "Cloud-Based Push-Styled Mobile Botnets: A Case Study of Exploiting the Cloud to Device Messaging Service," in *28th Annual Computer Security Applications Conf.*, 2012.

[12] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani, "Energy Consumption in Mobile Phones: A Measurement Study and Implications for Network Applications," in *9th ACM SIGCOMM Conf. on Internet Measurement*, 2009.

[13] R. Krashinsky and H. Balakrishnan, "Minimizing Energy for Wireless Web Access with Bounded Slowdown," *Wireless Networks*, vol. 11, pp. 135–148, 2005.

[14] A. Sharma, V. Navda, R. Ramjee, V. N. Padmanabhan, and E. M. Belding, "Cool-Tether: Energy Efficient On-The-Fly WiFi Hot-Spots Using Mobile Phones," in *5th Int'l. Conf. on Emerging Networking Experiments and Technologies*, 2009.

[15] D. Hasenfratz, A. Meier, M. Woehrle, M. Zimmerling, and L. Thiele, "If You Have Time, Save Energy With Pull," in *8th ACM Conf. on Embedded Networked Sensor Systems*, 2010.

All online references in this paper were last accessed and validated in October 2013.