CalendarCast: Setup-Free, Privacy-Preserving, Localized Sharing of Appointment Data

Florian Echtler
Mobile Media Group
Bauhaus-Universität Weimar, Germany
florian.echtler@uni-weimar.de

ABSTRACT
We introduce CalendarCast, a novel method to support the common task of finding a suitable time and date for a shared meeting among co-located participants using their personal mobile devices. In this paper, we describe the Bluetooth-based wireless protocol and interaction concept on which CalendarCast is based, present a prototypical implementation with Android smartphones and dedicated beacons, and report on results of a user study demonstrating improved task performance compared to unaugmented calendars.

The motivating scenario for CalendarCast occurs quite often in a variety of contexts, for example at the end of a prior meeting or during ad-hoc conversations in the hallway. Despite a large variety of digital calendar tools, this situation still usually involves a lengthy manual comparison of free and busy time slots. CalendarCast utilizes Bluetooth Low Energy (BTLE) advertisement broadcasts to share the required free/busy information with a limited, localized audience, on demand only, and without revealing detailed personal information. No prior knowledge about the other participants, such as email addresses or account names, is required.

ACM Classification Keywords
H.5.2. User Interfaces: Mobile Interfaces; H.5.3. Group and Organization Interfaces: Synchronous Interaction; H.4.1. Office Automation: Time Management

Author Keywords
Bluetooth; Low Energy; BTLE; calendar; appointment; broadcast; anonymous; sharing

INTRODUCTION
As personal mobile devices have become ubiquitous, it is now also commonplace to use them for fixing appointments, especially in a work context. Consequently, we often see several people standing together, staring at their respective smartphones or tablets, while trying to find a common date and time for a meeting. This scenario repeatedly occurs at the end of a prior meeting, or during ad-hoc conversations in the hallway, and usually involves a back-and-forth with multiple proposals from one participant which are then rejected by one of the others until a common free time slot is found. Although cloud services, shared calendars and similar tools promise to ease this tedious task, they require a willingness to share calendar information with third parties, a commitment to using the same service from all participants, and particularly prior knowledge of the other persons. As soon as even one person does not use the same tool as the others or has just joined the group, the promised support breaks down and again needs to be replaced by manual comparison of calendars.

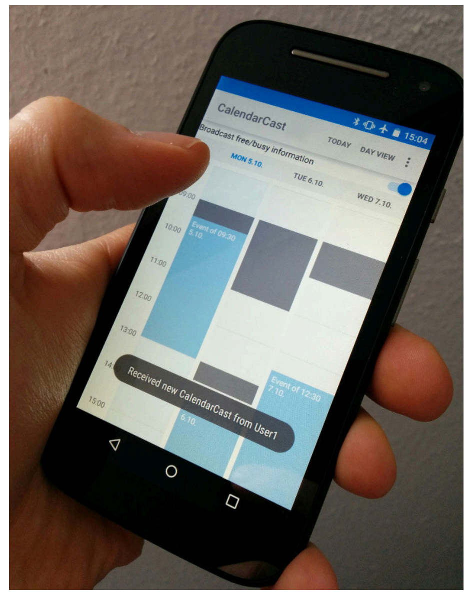

Figure 1. CalendarCast user interface. Own calendar entries are shown colored, while received broadcasts are shown as anonymous grey events in the background. Reception of a new broadcast is indicated by a popup message.

To address these issues, we introduce CalendarCast, a privacy-preserving, localized, and time-limited method of sharing free/busy information with co-located participants without requiring disclosure of personal information. CalendarCast uses Bluetooth Low Energy (BTLE) advertisement broadcasts with a very low range of 5-10 meters which can be sent and received by most modern smartphones and tablets. While these broadcasts are primarily designed to simply ad-

CHI'16, May 07 - 12, 2016, San Jose, CA, USA

ACM 978-1-4503-3362-7/16/05...$15.00
DOI: http://dx.doi.org/10.1145/2858036.2858100

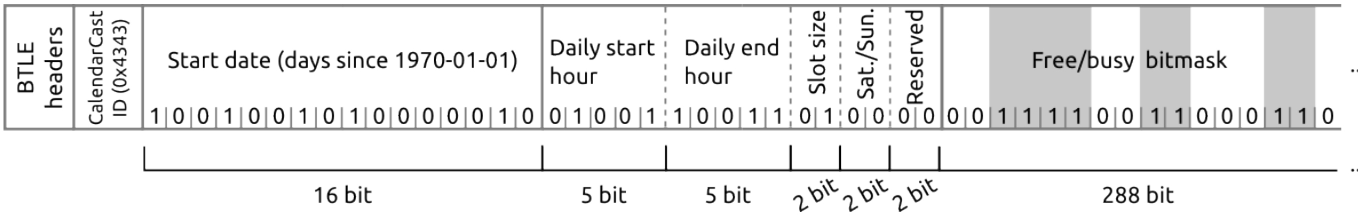

Figure 2. Bit-level description of CalendarCast message format (all numbers represented in network byte order, i.e. big endian format as suggested by [14]). This message is split and encapsulated in two BTLE broadcast frames.

vertise the presence and availability of a device, they offer limited space for additional user-specified data.

We re-purpose this reserved space to embed a bit pattern representing free and busy timeslots of the device's owner within a certain timeframe. Multiple devices can simultaneously broadcast and receive this information, thereby creating an ad-hoc shared calendar which is visible on all participating devices. Additionally, dedicated broadcasting devices can be used to distribute information about the availability of shared resources, such as a meeting room.

RELATED WORK

The topic of spontaneous interaction between co-located participants using their mobile devices has been explored by numerous researchers. Sarigol et al. present AdSocial [15], a mobile instant messaging application built around an ad-hoc wireless network. Shuttleworth et al. [17] propose using a similar network to facilitate an automated discovery process for appointments. However, both approaches require a shared local WiFi network which may have security implications or access restrictions and is therefore not always available.

Google Nearby [10] attempts to remove this restriction by providing a framework to determine nearby devices using a variety of techniques, such as WiFi, Bluetooth and inaudible sound. Nevertheless, this framework still requires a permanent Internet connection to a Google server to actually retrieve the list of peer devices. Similar functionality without the cloud connection requirement is offered by NFC+ [5] and Android Beam [9]; however, due to the use of NFC for the initial connection, both approaches are limited to two participating devices.

Do and Gatica-Perez use Bluetooth as a measure of proximity to discover the social context a person is in [8], while Davies et al. interact with public devices without a setup procedure by changing the device name broadcast by Bluetooth devices [7]. This technique has since been superseded by Bluetooth Low Energy. More recently, Aditya et al. present Encore [1] which employs BTLE broadcasts to create a personal log of encounters with people while at the same time creating event-specific secure encryption keys only available to those involved in the encounter.

Regarding the topic of calendar management, Schaub et al. show PriCal [16], a semi-public calendar display which preserves privacy by changing the detail level of the displayed information based on presence of other persons. Beard et al. present an early version of calendar overlays [3] to show the aggregate availability of multiple persons.

PhoneEar by Nittala et al. [12] uses inaudible sound embedded in regular audio streams to broadcast small data snippets to nearby devices. Tan et al. use "silence signatures" [18] to determine which mobile devices are in the same meeting without disclosing private information through audio recordings.

Despite this large body of existing work, we consider CalendarCast the first approach to combine setup-free interaction, preservation of privacy, and peer-to-peer techniques for sharing small information snippets such as calendar data.

CALENDARCAST

As described above, the main usage scenario for CalendarCast is finding a common free timeslot for an appointment among co-located participants, possibly including free/busy information for shared resources such as meeting rooms. We assume that users will quickly agree up front on a broad time range (e.g., "sometime next week", "in early November") within which the desired appointment should take place. Once this initial agreement has been made, each participant will then select individual constraints such as exact start day, duration of the time window and earliest/latest acceptable time of day.

Once these selections have been made, the devices will start broadcasting free/busy information for the chosen time window while at the same time listening for CalendarCast transmissions from peer devices. As soon as a broadcast from a peer is received, the contained information is integrated into the local calendar display as anonymous, untitled appointments which recede behind the users' own calendar entries (see also Figure 1). As more and more peer messages are received and integrated, the calendar fills up to display a union of all participants' free/busy information, thereby immediately highlighting which time slots are free for everyone. The final agreement on a specific time slot can then be confirmed verbally.

Low-Level Data Format

CalendarCast is based on Bluetooth Low Energy (BTLE) advertisement messages. These messages are periodically broadcast by BTLE devices with adjustable transmit power and interval, usually in the range of several 100 ms. The messages are unencrypted and can be received by all BTLE devices within range (usually 5-10 m) without prior pairing, in particular even by those which are simultaneously broadcasting themselves. One message can contain up to 31 bytes of payload [4] in the most basic "passive advertising" mode; when "active advertising" is used, a second message with 31 additional bytes will be broadcast upon request. These requests are performed automatically by listening devices, resulting in a total of 62 bytes which can be broadcast to nearby devices without setup or user intervention.

Of course, it would be possible to extend the amount of data using an additional protocol layer on top of the broadcasts which could employ, for example, sequence numbers to combine several successive messages. A generic implementation of this approach was presented by Corbellini et al. [6]. However, this solution would proportionally increase the duration until a full message has been received by peer devices and would also increase sensitivity to packet loss on the shared wireless channel, further increasing the time required to deliver all data. For these reasons, we decided to base CalendarCast on the small amount of data available in unmodified broadcasts. Nevertheless, even the comparatively tiny broadcast messages are already sufficient for our purpose of sharing free/busy information.

Of the 62 available bytes, between 10 and 13 bytes (depending on the BTLE implementation) are reserved for protocol-related header fields such as device capabilities and transmit power, while up to 7 additional bytes can be claimed for the "human-readable" device name which is always included in the broadcast. While this device name is often pre-set to a generic model identificator (such as "Galaxy" or "iPhone") and rarely changed by the user, this does not hinder the function of CalendarCast, as our focus is on creating an aggregate and *anonymized* calendar.

Due to these constraints, at most 42 bytes can be assumed to be available for CalendarCast data across all device types. Of these, 2 bytes are required for the CalendarCast identifier (0x4343, 'CC') and another 4 bytes for a CalendarCast-specific header describing the format of the bit pattern in the remaining 36 bytes or 288 bits of the message. This bit pattern represents a sequence of time slots, beginning at a start date specified in the header, and simply tells all receiving peers if the corresponding time slot is free (0) or busy (1).

An example message is illustrated in Figure 2. Here, the starting date is October 5th, 2015, i.e. 16713 days after January 1st, 1970. The daily time range is 9:00 - 19:00, and the slot size is 30 minutes (0 = 15min, 1 = 30min, 2 = 1h, 3 = 2h). This means that in this case, every day contains a total of 20 available slots. As weekends are excluded (bits for Sat./Sun. are set to zero), the 288 slot bits in the message can then cover a total of 14.4 days or just under three work weeks, which should usually be sufficient to find a common free timeslot. The end date is always implicitly specified by the number of slots available in the bitmap and the slot size.

User Interface

The user interface of CalendarCast mostly consists of a "classic" calender view with an additional button on top to start the broadcast. Upon tapping this button, the user is presented with a modal dialog in which the rough time window for the desired appointment can be adjusted (start date, daily start and end times, size of time slots).

	Sparse A/B	Dense A/B	Room
Time range	Oct. 5 - Oct. 16, 9am - 7pm		
Time slot duration	30 minutes (20/day)		
Total time slots	200 (weekends excluded)		
Busy time slots	40	80	80
Common free slots	133	67	n/a
# of individual events	22	24	28
Distance to first match	24	44	n/a

Table 1. Characteristics of all four test calendar pairs. Sparse/dense calendar pair A was tested with CalendarCast, sparse/dense pair B without.

Once this dialog has been completed, the calendar view is shown again and the broadcast of free/busy information starts in the background. At the same time, the device also starts listening for broadcasts with the custom CalendarCast identifier. As soon as a compatible broadcast is received from a previously unknown device, the contained free/busy information is added to the calendar view as "background events", i.e. untitled events in muted colors which recede behind the user's own events in the foreground. These events serve as a visual aid to quickly locate timeslots that are available to all participants in a kind of "focus & context" view, with the user's own events being in *focus* and the background events from other users providing the *context*. Ad-hoc modification of calendar entries is of course still possible, and will quickly propagate to other users' displays.

EVALUATION

We evaluated a prototypical implementation of CalendarCast on Android smartphones. We used Motorola *moto e* (2nd generation) devices running Android 5.0, as this hard- and software combination is known to support simultaneous sending and receiving of BTLE broadcasts [13]. Our hypotheses were that CalendarCast will enable participants to find a suitable time and date for a planned appointment faster than using manual comparision of calendars, and that this effect will be more pronounced for more complex search tasks, e.g. on calendars with more events which have less overlap of free time slots. We intentionally excluded usability-related questions from our study and focused on task completion times, as any calendar interface would include numerous confounding variables (e.g. number of days in view) that would be difficult to control for. Consequently, we did also not yet compare CalendarCast to existing calendar tools.

Our test setup consisted of two *moto e* smartphones preloaded with test calendars in a custom app, a BTLE dongle[1] for broadcasting availability data of a hypothetical meeting room, and a paper calendar representing the meeting room's door calendar (showing the same data as broadcast by the dongle). Participants were tested in pairs of two, and given the task of finding two common appointments within a given time window (first two weeks of October 2015) based on the

[1]based on Nordic nRF51822 BTLE SOC

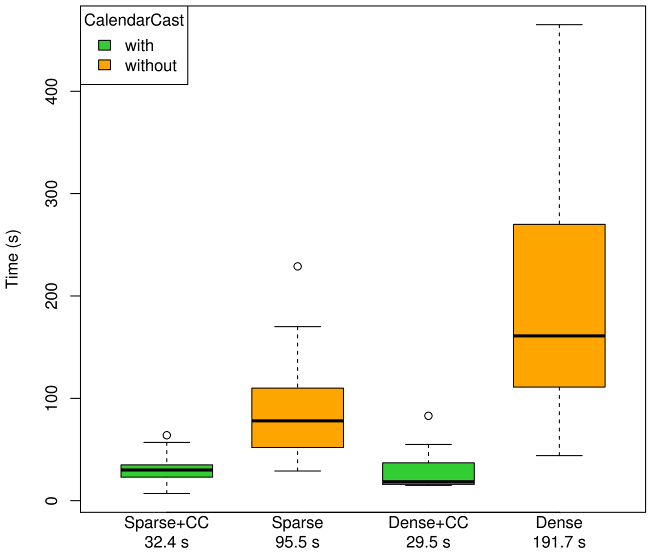

Figure 3. Task performance (boxes show 2nd and 3rd quartile, with median highlighted). Mean value is indicated below each condition.

available calendar data, including the meeting room. As additional constraints, both appointments had to span two hours and occur on the same weekday and time (i.e., a recurring appointment) to simulate a non-trivial search task in a realistic setting. Time was taken from the simultaneous start of the app on both devices to the confirmed agreement on a time slot.

We tested four conditions in a within-subject design using balanced Latin square order [11]: sparsely populated calendars and densely populated calendars, both with and without CalendarCast support. The time window was already pre-set in the app to match the test calendars. The calendar data was different for each scenario, but contained the same amount of overlap, i.e. the same number of possible time slots. Distance from the start of the week to the first suitable event was identical for the two pairs of sparse and dense calendars, respectively, to ensure comparable search times. Details of the different data sets are described in table 1.

Our test was completed by 20 participants recruited from students and researchers of our university (N = 10 teams in total, average age 24.3 years, 7 female, 13 male). Test subjects did not receive any compensation for their participation, and consented to the anonymized and aggregated use of their recorded data. On average, subjects reported to use smartphones "several times daily" and digital calendars at least "several times weekly".

Although the total number of samples is small, the results clearly support our two initial hypotheses as illustrated in Figure 3. Without CalendarCast, the average task completion time was 95.5 seconds for the sparse calendar and 191.7 seconds for the dense calender. The ratio between these two results (1:2.01) also roughly corresponds to the ratio between the distances to the first match (24:44 = 1:1.83, see also table 1). When using CalendarCast, however, task time dropped to 32.4 seconds for the sparse and 29.5 seconds for the dense calendar, a reduction by a factor of 2.9 and 6.5, respectively.

A tentative statistical analysis using the Shapiro-Wilk test shows that the data is likely not normally distributed ($p < 0.05$ for two conditions), and a subsequent Wilcoxon Signed-Rank test confirmed that both in the sparse ($p = 0.0078$) and dense ($p = 0.0039$) conditions, a statistically significant difference between the variants with and without CalendarCast exists. Interestingly, when testing for difference between the two CalendarCast-based conditions, the result suggests ($p = 0.3131$) that *no* difference exists. This implies that using CalendarCast reduces the task time to a constant value, regardless of how much overlap between the calendars exists.

DISCUSSION & OUTLOOK

We have presented CalendarCast, a privacy-preserving, localized, and time-limited method to share calendar data with one's immediate surroundings without requiring prior setup. Our evaluation confirms that CalendarCast improves task completion time when trying to find a common free time slot for an appointment among co-located participants, particularly for more complex search tasks with multiple constraints.

Although our prototype was developed and tested on Android, similar techniques could also be applied on iOS-based devices. However, it is worth noting that iOS as of v9.0 offers only limited control over the low-level message contents. Currently, it is only possible to include UUIDs into the broadcast data [2], i.e. fixed-size values of either 16 bit or 128 bit length. Consequently, while it is still possible to create CalendarCast messages using custom UUID values, the effective payload size will be lower on these devices.

While the app developed for the evaluation employed an internal database to provide the calendar data for the different test scenarios, a future deployment "in the wild" would use standardized mobile OS interfaces to access the users' own calendar entries and build CalendarCast messages from them.

We consider several possible extensions to CalendarCast which we will investigate in future experiments. The first extension involves the semi-automated broadcast of appointment proposals (either auto-selected or initiated by one user) to the other participants, who then only have to confirm the proposed time slot after specifying the acceptable time frame.

Another possible extension would be to re-purpose the bitmask used by CalendarCast to represent a set of interests instead; in a venue such as a large conference, attendants could periodically broadcast the specific topics they are interested in and be notified of persons nearby who share a noticeable number of topics. An even more generic approach would be to allow sharing of arbitrary data items by using compact representations, e.g. a PGP global key ID for contact information or a shortened URL for media data.

ACKNOWLEDGMENTS

We thank Raphael Wimmer, Stefanie Wetzel, and Sven Bertel for their valuable comments, and Felix Schmidt for his help in conducting the study. This work is in part supported by BMBF through grant T_209_M_SB.

REFERENCES

1. Paarijaat Aditya, Viktor Erdélyi, Matthew Lentz, Elaine Shi, Bobby Bhattacharjee, and Peter Druschel. 2014. EnCore: Private, Context-based Communication for Mobile Social Apps. In *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '14)*. ACM, New York, NY, USA, 135–148. DOI: http://dx.doi.org/10.1145/2594368.2594374

2. Apple, Inc. 2015. Peripheral Role Tasks - Advertising Your Services. (2015). https://developer.apple.com/library/prerelease/ios/documentation/NetworkingInternetWeb/Conceptual/CoreBluetooth_concepts/PerformingCommonPeripheralRoleTasks/PerformingCommonPeripheralRoleTasks.html#//apple_ref/doc/uid/TP40013257-CH4-SW5 Retrieved September 25, 2015.

3. David Beard, Murugappan Palaniappan, Alan Humm, David Banks, Anil Nair, and Yen-Ping Shan. 1990. A Visual Calendar for Scheduling Group Meetings. In *Proceedings of the 1990 ACM Conference on Computer-supported Cooperative Work (CSCW '90)*. ACM, New York, NY, USA, 279–290. DOI: http://dx.doi.org/10.1145/99332.99361

4. Bluetooth SIG. 2014. Core Specification 4.2. (2014). Retrieved September 25, 2015.

5. Kuang-Ming Chen, Yu-Cheng Liou, and Mike Chen. 2011. NFC+: NFC-assisted Media Sharing for Mobile Devices. In *Proceedings of the 13th International Conference on Ubiquitous Computing (UbiComp '11)*. ACM, New York, NY, USA, 575–576. DOI: http://dx.doi.org/10.1145/2030112.2030220

6. Giorgio Corbellini, Stefan Schmid, and Stefan Mangold. 2015. Two-Way Communication Protocol Using Bluetooth Low Energy Advertisement Frames. In *Proceedings of the 1st International Workshop on Experiences with the Design and Implementation of Smart Objects (SmartObjects '15)*. ACM, New York, NY, USA, 19–24. DOI: http://dx.doi.org/10.1145/2797044.2797049

7. Nigel Davies, Adrian Friday, Peter Newman, Sarah Rutlidge, and Oliver Storz. 2009. Using Bluetooth Device Names to Support Interaction in Smart Environments. In *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services (MobiSys '09)*. ACM, New York, NY, USA, 151–164. DOI: http://dx.doi.org/10.1145/1555816.1555832

8. Trinh Minh Do and Daniel Gatica-Perez. 2013. Human Interaction Discovery in Smartphone Proximity Networks. *Personal Ubiquitous Comput.* 17, 3 (March 2013), 413–431. DOI: http://dx.doi.org/10.1007/s00779-011-0489-7

9. Google, Inc. 2015a. Android Beam. (2015). Retrieved September 25, 2015.

10. Google, Inc. 2015b. Nearby. (2015). Retrieved September 25, 2015.

11. I. Scott MacKenzie. 2013. *Human-Computer Interaction: An Empirical Research Perspective* (1 edition ed.). Morgan Kaufmann, Amsterdam.

12. Aditya Shekhar Nittala, Xing-Dong Yang, Scott Bateman, Ehud Sharlin, and Saul Greenberg. 2015. PhoneEar: Interactions for Mobile Devices That Hear High-frequency Sound-encoded Data. In *Proceedings of the 7th ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS '15)*. ACM, New York, NY, USA, 174–179. DOI: http://dx.doi.org/10.1145/2774225.2775082

13. Radius Networks. 2015. Android Beacon Library. (2015). Retrieved September 25, 2015.

14. John Reynolds and John Postel. 1994. RFC1700 - Assigned Numbers. (1994). Retrieved September 25, 2015.

15. Emre Sarigöl, Oriana Riva, Patrick Stuedi, and Gustavo Alonso. 2009. Enabling Social Networking in Ad Hoc Networks of Mobile Phones. *Proc. VLDB Endow.* 2, 2 (Aug. 2009), 1634–1637. DOI: http://dx.doi.org/10.14778/1687553.1687611

16. Florian Schaub, Bastian Könings, Peter Lang, Björn Wiedersheim, Christian Winkler, and Michael Weber. 2014. PriCal: Context-adaptive Privacy in Ambient Calendar Displays. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '14)*. ACM, New York, NY, USA, 499–510. DOI: http://dx.doi.org/10.1145/2632048.2632087

17. Saxon Shuttleworth, Rebecca Chan, Julka Trajic, and Henricus Gerardus Spaay. 2003. System and method for synchronous peer-to-peer appointment scheduling facilitation. (July 2003). CIB: G06Q10/10; (IPC1-7): G06F17/60.

18. Wai-Tian Tan, Mary Baker, Bowon Lee, and Ramin Samadani. 2013. The Sound of Silence. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems (SenSys '13)*. ACM, New York, NY, USA, 19:1–19:14. DOI: http://dx.doi.org/10.1145/2517351.2517362