Dynamic group management with Bluetooth Low Energy

D. Giovanelli, B. Milosevic, C. Kiraly, A.L. Murphy, E. Farella
ICT Center, Fondazione Bruno Kessler (FBK), Trento, Italy.
Email: {dgiovanelli, milosevic, kiraly, murphy, efarella}@fbk.eu

***Abstract*—The flourishing ecosystem of the Internet of Things (IoT) provides the enabling technologies for the development of smart cities. In intelligent urban settings, integrated and sustainable mobility plays a key role, especially for children, whose ability to move independently throughout their neighborhood is fundamental, both for themselves and their community. Our case study comes from a stepping stone toward independent mobility, namely a group of children walking to school under adult supervision. For this scenario, we offer a tool based on wireless, wearable nodes that supports adults to safely form and manage a group of children. We describe the application requirements, generalizing them beyond this particular use case. We also offer a prototype implementation that exploits Bluetooth Low Energy connectionless communication to build bi-directional information exchange for group management. Simulations and pilot tests validate the proposed approach in terms of scalability, network performance and battery life.**

I. Introduction

Recent technological innovations such as miniaturized networked sensing devices, widespread connectivity and advanced but affordable remote storage and processing services have paved the way for the Internet of Things (IoT) [1]. Taken together, these offer a rich ecosystem that enables the development of smart cities and the diffusion of innovative applications. Integrated mobility and the development of IoT systems and services for sustainable, efficient and independent mobility within the future smart cities is an active and inspiring application area [2, 3, 4].

In this smart urban scenario, pedestrian mobility and the management of dynamic groups of people represent an important corner stone. We consider groups such as organized tours, families with children, as well as informal groups of friends. We take particular inspiration from a scenario with both educational and societal impact, namely the safe and secure mobility of children throughout their neighborhoods. Within the CLIMB project, we are exploring ways that technology can be used to support teachers, parents and students in the goal of increasing both independence and safety as well as encouraging sustainable mobility. Our multi-pronged solution incorporates wearable IoT devices as well as gamification to increase motivation and participation.

This paper focuses on a first step toward the eventual goal of independent mobility, namely supervised mobility of a group of children walking from their homes to the school, known as a walking bus. This scenario, with a few adults responsible for multiple children, maps directly to school outings where the responsibility lies with the teachers. Our focus lies with the IoT angle, developing a group management system for individuals carrying wireless, mobile devices. We support *(i)* group formation, e.g., tracking which children are aboard the walking bus, and *(ii)* membership monitoring, e.g., raising an alarm if a child wanders too far from the *driver* of the walking bus. Our solution uses small, autonomous nodes carried by all participants, as well as a smartphone and the accompanying app carried by a single parent monitor.

In this paper, we first outline related efforts from the literature then focus on the application requirements for the walking bus and how they generalize to other group scenarios. We move on to describe the design and implementation of our solution that uses the connectionionless capabilities of the Bluetooth Low Energy (BLE) standard to build a flexible, efficient group monitoring service. The group management itself is built using only broadcast messages, but it uses this to build bi-directional information exchange between the parent and child nodes to increase the reliability of the system.

Our experimental evaluation analyzes performance in terms of network latency and delay, system scalability and battery life. Moreover, we present results from a controlled test of the walking bus scenario in an outdoor, real-life environment.

II. Related work

Group management through wireless wearable nodes has been tackled by the research community in the past using a variety of technologies. The work presented in [5] focuses on the problem of decentralised group management in general, from a more theoretical perspective. Three different protocols are introduced for the dissemination of group membership information. The protocols are compared in a simulated environment in terms of energy consumption, accuracy, and latency, but no real world implementation is provided.

Other works take a more pragmatic approach concentrating on the problem of child kidnapping, and designing systems for the tracking of children through wearable wireless devices and their physical vicinity to trusted GPS enabled devices. Lee et al. in [6] propose a system based on the de-facto standard TelosB WSN node. Kids wear TelosB devices operating as beacons, while monitoring nodes are composed of a TelosB node with a Bluetooth module that attaches to a smartphone. The system is validated in a very limited setup of 5 nodes. Recently, commercial start-ups, e.g., Lineable [7], have also begun to address related problems, such as child safety. The Lineable solution is based on a wristband used as a BLE beacon. Beacons are detected by smartphones, and presence information is shared among the participating smartphones through the cloud. None of the above, however, supports bi-directional communication in large groups, a technological advantage which we exploit in our solution.

III. Requirements

Recent IoT technologies offer several possibilities for the core of a system to support the management of a group of people in motion. In particular, miniature devices that are capable of dynamically forming wireless networks and sub-networks and estimating their distances represent ideal candidates. Here, we describe the requirements of a group management system composed of low power wearable wireless modules, able to transmit and receive small amounts of data independently from the chosen protocol.

The requirements for the proposed application can be summarized as follows:

- **Discover new users when in range** and, if they are part of the desired group, add them to a current *friend list*.
- **Detect if one of the current members separates from the group**, notifying both the disconnected user and the group manager and possibly the rest of the group.
- **Spatial localization** of the group members relative to the rest of the group or to the group manager. There is no need for highly accurate localization; a coarse, relative estimate is sufficient.
- **Bi-directional and multi-hop communication** to allow the manager and different group members to sense each other's presence even if they are not in direct contact due to occlusions or interference.
- **Latency and scalability** are interconnected as a large number of nodes with frequent transmissions will result in a large number of concurrent transmissions causing packet collisions and data loss. We set the maximum number of nodes that can be in the communication range to 150 and the maximum tolerated packet loss to 10%. The maximum discovery latency is 5 seconds.
- **Long battery life** to avoid frequent recharging.
- **Compatibility with established technologies** to allow the system to integrate with existing devices and infrastructure.

IV. System description

A. General overview

To implement a group management system, we define a Leader-Member architecture, with one node acting as the group leader and all others taking the member role. In our network, the Leader also acts as a gateway towards the external world for data exchange and user interaction. Member nodes are intended to be autonomous and require minimal user interaction. While this architecture naturally maps to supervised groups, e.g., with the teacher taking the Leader role, it can easily be adapted to peer groups, e.g., by automatically electing the Leader during group formation.

TABLE I: Matrix of possible Member node's states

		Wireless network state	
		0	1
Group state	0	BY_MYSELF	CHECKING
	1	ALERT	ON_BOARD

Given this architecture, the Leader node will track and manage the state of the group, while the Member nodes assume different states depending on their localization and network state. Table I summarizes the proposed states for a Member node, which is a combination of its wireless link state (i.e. if it is in contact with the Leader) and of its group state (i.e. if it is part of a group). The four resulting possibilities are:

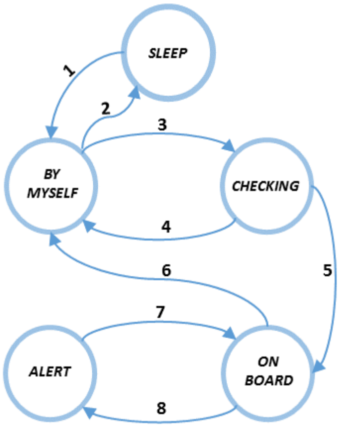
- **BY_MYSELF:** The Member node is far from any Leader and it is not part of any group.
- **CHECKING:** The Member is in direct contact with a Leader node, but it is still not part of its group.
- **ON_BOARD:** The Member is in direct contact with a Leader node and it is part of its group. In this case the Leader is monitoring its presence.
- **ALERT:** The Member is part of a group but, at the moment, it is out of range and not in direct contact with its Leader.

Since the Leader supervises the network, it directs the state transitions of its Members, with the exception of the ALERT state. In this case, when the Member is out of communication range with its Leader, the ALERT must be autonomously detected at both sides. For increased reliability, each state transition is requested by the Leader and subsequently acknowledged by the Member. This approach requires bi-directional communication and ensures correct synchronization of the group nodes states.

The proposed system behavior is the following:

- At power on/wake up a Member is in BY_MYSELF and remains in this state until it comes in contact with a Leader.
- When the Leader discovers a new Member, it requests that Member to move to the CHECKING state, which will be performed and acknowledged.
- The Leader checks if such node is part of its group, which may happen according to a set of policies (e.g. accepting all nodes, checking a list or requesting user interaction).
- To add the Member to the current group, a state transition to ON_BOARD is requested from the Leader and acknowledged by the Member. From now on, both the Leader and the Member will monitor each other's presence.
- If the Member goes out of the Leader's range and stops receiving its communications for a certain time both nodes will trigger the transition to ALERT and will notify the user.

The complete set of states and the events that trigger their transitions are depicted in Figure 1. We also consider a SLEEP state when a node is not in use.



	Event
1	Timer or button event
2	Timer or button event
3	The Member node enters in the Leader communication range
4	The Member goes out of Leader range or the Leader reject to monitor the Member
5	The Leader adds the Member to its friend list
6	The Leader stops monitoring the Member
7	The Member returns in Leader communication range
8	The Member no longer hears the Leader

Fig. 1: Member state machine along with the high level events that trigger the state transitions.

B. Network implementation

Our group monitoring system is based on the exchange of wireless broadcast messages among proximate nodes. In a nutshell, the group Members periodically announce their presence. The Leader listens for these announcements and forms a local group membership list. To control the state changes of the Members, as outlined in the previous section, the Leader periodically announces this list, which is received by the Members, who update their states.

Our current implementation is built on top of BLE, from which we utilize the *BLE advertiser* and *BLE observer* modes for respectively sending and receiving broadcast messages without establishing connections. BLE natively supports periodic advertisements, at the so-called *advertise interval*, T_{AI}, in the range of 20 ms to 10 s. This forms the core of our Member behavior, where the advertisement message of the Members contains the 8-bit node identifier and the application state, which is sent every $T_{AI,m}$. Immediately after each transmission, the Member switches into the observer state to listen for other advertisement packets. The Leader also uses periodic advertisements to announce its state, namely a list of the nodes in its group, and the application state that the Leader intends for them to switch to. We set this period to $T_{AI,l}$. The Leader switches to listening between advertisements.

It is worth noting that the payload of the BLE advertisement is 31 bytes long, limiting the size of the membership list to 9 ID-state pairs. To handle larger groups, the Leader cycles through the Member list in subsequent advertise packets. For example, with a group size of 12 Members, the Leader will announce the first 9 in one advertise packet, then after $T_{AI,l}$ it will announce the remaining 3 and will repeat the first 6. This scheme offers a deterministic communication latency, dependent on the maximum size of the group.

To increase usability, we extend this core solution in three key ways. First, we note that the Members must actively listen for advertisements from the Leader in order to update their states. While listening, Members also overhear the advertisements of other Members. In this first extension, we simply save this information to augment knowledge about node connectivity. Specifically, each Member node maintains a list of its neighboring nodes, storing their ID and the RSSI values of the received packets. This information is sent along with the ID-state pair as part of the Member's advertisement. As we have the same limit to pack at most 9 pairs of information in each advertisement, we employ the same mechanisms to rotate among the neighbor information in subsequent packets.

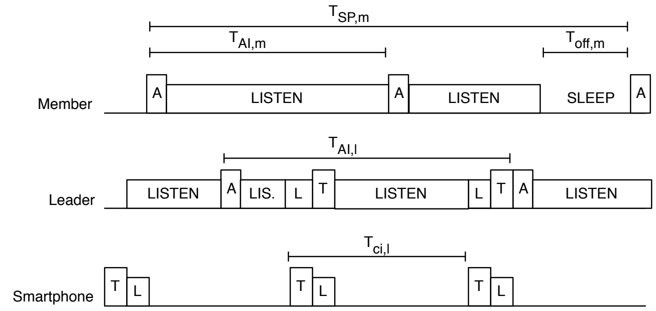


Fig. 2: Schematic of the communication pattern. Note that this is not drawn to scale. Specifically transmission events have a very short duration.

Second, while Member nodes are required to listen to receive their updated states from the Leader, we must also consider the battery consumption of these devices. Therefore, we choose to put the radio to sleep for slightly less than one advertising period $T_{AI,l}$ every two periods, yielding the periodic behavior shown in Figure 2 in which the node wakes up, transmits its state and the connectivity information of some of its neighbors, then either listens for the whole period or listens for slightly more than half the period then goes to sleep. Clearly when the node is sleeping, it does not hear the advertisements of the other nodes, and it may miss the Leader requesting it to change state. Nevertheless, as the node is likely to hear the subsequent transmission, the overall, correct behavior is maintained, albeit with a delay acceptable in our target applications.

Finally, we extend the behavior of the Leader node to allow communication with a proximate, more powerful device, e.g., a smartphone, useful for user interaction. For this, we establish a connection between the Leader device and the smartphone, with the Leader acting as a *BLE peripheral*. Information between these devices is exchanged at a period of $T_{CI,l}$. This is shown schematically in Figure 2 as short listening (L) and transmission (T) events between the Leader and the smartphone.

V. Experimental Evaluation

A. Experimental Setup

The prototype of the proposed system is based on the CC2650 chip by Texas Instruments [8]. It is a System-on-Chip, which includes all the RF circuitry and a Cortex M0 core dedicated for the lower layers of the BLE stack implementation and one additional Cortex M3 core for user application and higher BLE stack layers. For the test deployment, we used the SensorTag development kits, which include the CC2650, a set of sensors (inertial, temperature, light), two application LEDs

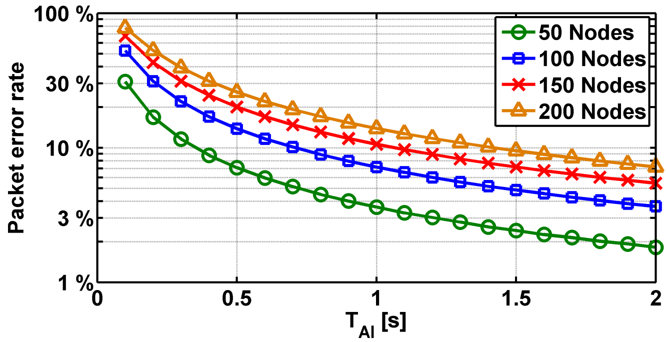


Fig. 3: Percentage of corrupted packets with respect to the advertise interval (t_{AI}) for 50, 100, 150 and 200 nodes in range.

and buttons on a $32 \times 42mm$ board powered by a coin battery (CR2032). Both Leader and Member nodes are implemented using the same hardware, albeit with different firmware. The final implementation includes a Nexus 9 tablet that runs an application on the Android 5.1.1 operating system.

B. Network density and timings

As reported in the previous Sections, the advertise interval T_{AI} sets the trade-off between the network's latency and the supported node density. The latter dictates the maximum number of co-existing nodes and is limited by the loss of advertising packets due to collisions. To calculate the T_{AI} that guarantees the target performance, we simulated the network to estimate the Packet Error Rate (PER) given a number of nodes and a T_{AI}. We simulated one advertise period for each node randomizing the time at which the packet is sent and considering a packet corrupt any time any overlap occurred. For each configuration, the simulation was repeated 10^5 times and the average PERs are reported in Figure 3. It should be noted that this simulation is hardware-independent and is valid for any BLE advertiser with a passive BLE observer.

Because we have only 10 actual nodes in our laboratory for testing, we validated the simulation results by shortening the advertising intervals to maintain similar density for the number of packets transmitted in a given time interval. We performed our experimental validation in our office laboratory environment, with nodes uniformly distributed in a $0.5m^2$ area, placing them well within communication range. We experimented with 1, 2, 5 and 10 transmitting Member nodes and with 20, 40, 60 and 80 ms advertise intervals. The nodes were programmed to broadcast 5000 packets and a packet sniffer logged all received data, allowing us to compute the average PER. Note that the average advertise interval is 5 ms longer than the nominal one due to the random [0...10] ms delay added by the BLE protocol stack.

The results of our tests are shown in Figure 4, where we observe that the simulation closely matches the experimental validation. The variations can be attributed to the fact that the simulation does not consider some hardware characteristics such as channel switching by the receiver, concurrent accesses to the radio peripheral and external interferences typical of an office scenario (e.g. wi-fi, mobile phones). These side effects are particularly visible in the experimental PER with only one transmitting node (Figure. 4-a). In this experiment, no packet collisions occur but we measured an average of 0.2% of lost packets.

Considering the network simulation and its experimental validation, we set the Member advertising interval to 1 s ($T_{AI,m}= 1s$), which leads to an average packet error rate of 10.6% with 150 nodes. Since the Leader node must track and manage all the other nodes, we lowered its advertising interval to 0.625 s ($T_{AI,l}= 0.625s$). The transmission of advertising packets employs the radio for 3 ms and the remaining time the nodes are free to switch to listening for incoming packets using the BLE observer mode. To increase energy efficiency, we duty cycled the receiver with 1.25 s of activity every 2 s. This solution gives us a Member listening duty cycle of 62.5%, which reduces the power consumption of the device and leads to the timing profile illustrated in Figure 2.

Time multiplexing is used to manage concurrent transmission and reception accesses to the radio peripheral inside the CC2650. Therefore, even if we want the Leader's receiver to be always-on and ready to receive the advertise packets, this is not achievable due to the need to transmit advertise packets. In our setting, we measured the Leader's listening duty-cycle as 93.6%. The remainder of the time (6.4%) the Leader's radio is involved in the transmission of advertise packets or in exchanging data with the tablet through the BLE connection.

C. Discovery latency and round-trip delay

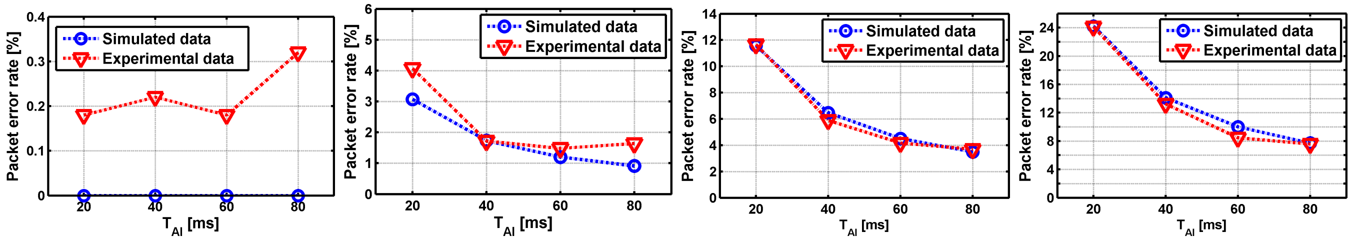
To evaluate the average time for the leader to discovery a member node, we set up one Member and we connected the Leader to the tablet via BLE. This setup recreates our test base in which the Leader is paired with a smartphone for system configuration and interaction.

We define the discovery latency (t_D) as the time from the instant a Member device enters the Leader's range until the moment a notification of its discovery reaches the smartphone app. The discovery latency can be seen as a one-directional network delay, from a Member to the smartphone. In the worst case (t_D^+) when the receiver is always on, this is given by:

$$t_D^+ = t_{CI} + 3t_{AI,m}^+ \tag{1}$$

where $t_{CI} = 220ms$ is the connection interval for the BLE connection between the Leader and the smartphone and $t_{AI,m}^+ = 1.01s$ is the worst case advertise interval for a Member, leading to $t_D^+ = 3.25s$.

To limit the probability of repeated packet loss due to radio artifacts, the connection and the advertise events (which occupy the Leader's radio) are executed at non-multiple time intervals ($t_{CI} = 220ms$, $T_{AI,l}= 625ms$). This ensures that the maximum number of consecutive lost packets is limited to two. In fact, the worst case is when the Member's first packet is transmitted during a Leader's connection event and the second one is transmitted during a Leader's advertise event. In such case, the subsequent packet will be received by the Leader, since it will be broadcasted in a Leader's listening interval (see Figure 2).



(a) 1 Member. Irregularities due to the small y-axis scale.

(b) 2 Members

(c) 5 Members

(d) 10 Members

Fig. 4: Packet error rate obtained from simulation (blue/circles) and from experimentation (red/triangles) using different numbers of Member nodes. Note: scale varies across figures.

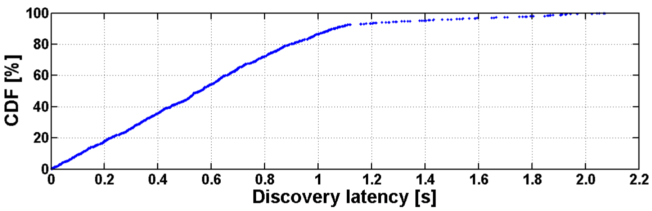


Fig. 5: Cumulative distribution function (CDF) for a new Member entering in the communication range of the Leader.

To experimentally measure the discovery latency we used the approach of [9], in which the process of entering the node's communication range is simulated for practical reasons. Specifically, the Member node being examined is considered to be out of range until a randomly chosen time (t_0). Then, the time when the node is detected, denoted as the contact time t_c, is identified, and the discovery latency is calculated as $t_D = t_c - t_0$. This sequence is repeated 1000 times and the obtained cumulative distribution function (CDF) is showed in Figure 5. As only one Member was active, this evaluation is of a collision-free configuration. The results show that in 93% of the cases the node is discovered within $1.12s$. The reminder of the cases (approximately 7%) are detected within $2.2s$, which is below the calculated value of t_D^+, and represent the cases where the first packet is lost due to the Leader's radio transmission multiplexing, but the second one is correctly received. While this test never observed two consecutive packets losses, one cannot exclude pathological cases when packets continue to collide. In other words, practically speaking, we have identified a maximum discovery time which is valid in a collision free environment, but theoretically, longer discovery latencies are possible in a realistic use case environment. These measurements closely match our earlier estimates.

The round-trip delay (t_{RT}) is the time for a packet, originating in the app, to reach a Member node and return back to the app. This is important since it represents a node state change request and its acknowledgment and it strongly depends on the network timings. Summing up the time intervals and delays the notification encounters through the network, it is possible to calculate the worst (t_{RT}^+) and best (t_{RT}^-) cases:

$$t_{RT}^+ = 2t_{AI,l}^+ + t_{off,m}^+ + 2t_{AI,m}^+ + 3t_{CI} \quad (2)$$

$$t_{RT}^- = 2t_{AI,l}^- + t_{AI,m}^- \quad (3)$$

Here, $t_{off,m}^+ = 770ms$ is the worst delay caused by the sleep within the scan period ($T_{SP,m}$, see Figure 2), $t_{AI,m}^+ = 1.01s$ and $t_{AI,l}^+ = 0.635s$ are the worst case advertise intervals for the Member and Leader nodes and the respective best cases are $t_{AI,m}^- = 1s$ and $t_{AI,l}^- = 0.625s$. With the chosen parameters we obtain $t_{RT}^+ = 4.72s$ and $t_{RT}^- = 2.25s$. Even with the previously described Leader's receiver duty cycling, during our experimental validation we obtained $t_{RT,MAX} = 4.54s$ and $t_{RT,min} = 2.68s$, which is in between t_{RT}^- and t_{RT}^+, the average measured value is $t_{RT,avg} = 3.37s$.

D. Memory requirements

Another key factor to allow the system to manage a large number of nodes is the memory footprint on each device. The Leader node must track all of the Member nodes with their states and the Member nodes also track their neighboring nodes. Our device is equipped with $20kB$ of SRAM. The application, the BLE stack and the operating system use $17.4kB$, leaving $2.6kB$ for storing node information. To store information about each neighboring node we require 35 Bytes of memory, hence each node can handle and track up to 76 devices. While this is smaller than the maximum value of 150 established in our requirements, we can imagine reaching larger group sizes by managing multiple co-existing groups with different Leaders.

E. Power consumption

To estimate the battery life of the proposed implementation, we applied the following consumption model:

$$t_{BL} = \frac{C_{batt}}{\delta I_A + (1-\delta) I_S} \quad (4)$$

where t_{BL} is the battery life in hours, C_{batt} is the battery capacity in mAh, I_A is the average active current and I_S is the sleep current. We assume that the device is not constantly used and δ is its usage duty cycle (i.e. if the device is used only one hour per day $\delta = 1/24$).

For our platform and the chosen parameters $I_A = 3.6mA$, $I_S = 100\mu$A and $C_{batt} = 225$mAh. The resulting expected battery life, if the device is used for one hour a day, is $t_{BL} =$ 900h (or 37.5 days). This is an optimistic estimate since it does not take into account the processing each node must perform, which slightly increases the average current absorption, and the battery's internal resistance. In an experimental test in the same conditions, we measured a battery life of 24 days.

F. Case study: Walking bus

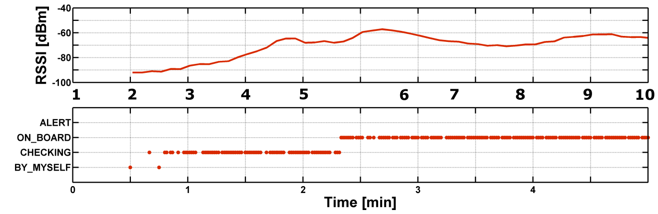
To validate our system, we performed an experiment simulating a walking bus with our research staff members. For one week, every day at the same hour, 10 people met and walked together on a 400 m path using our system to manage the group. The outcome of this experiment was the log of each Member's state, the RSSIs among all node pairs and the GPS log from the leader's smartphone. As example, Figure 6a reports information for one Member during one of the performed walks, summarizing the Member to Leader RSSI and the associated Member state. In Figure 6b part of the Leader's GPS trace is reported and superimposed on the map of the route. The numbers identify the sequence of the walk and align the spatial information to the collected signals.

During this walk, the Member in focus was waiting in position 5 in Figure 6b (on the right side of the picture) and the Leader user approached from the left, following the numerical sequence. Once the Leader reached the Member, he checked him in (the state changes to ON_BOARD, as seen in Figure 6a just before minute 2.5) and they continued together following the outlined path. The Member to Leader RSSI has been filtered applying a 30s sliding window with a 25s overlap. The test was driven by the need to observe the network in an outdoor environment and to identify possible issues or bugs. Further testing will be performed focusing on the system performance and on the estimation of distances between group members.

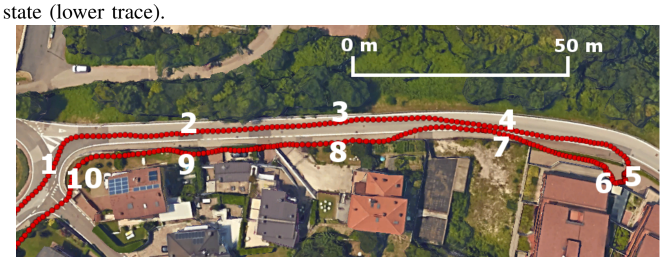
VI. Conclusions and Future Work

We developed and evaluated a system for group management using Bluetooth Low Energy. The main difference between our system and those based on beacons is that our Members are also configured to receive, unlike other systems that only transmit. This allows monitoring of Members that are not in direct contact, but are within two hops from the Leader. We clearly see opportunities to further reduce the consumption at each node, but we note that our implementation today offers a clear proof of concept that such systems are feasible for Smart City scenarios, especially considering our use of the standard BLE protocol, which is supported by a majority of smartphones on the market at the time of publication.

Starting from the application requirements, we developed the system design, tuning the parameters to achieve our target performance. Further, we performed extensive experimental validation to verify the effectiveness of the proposed approach. Our system has been tested "in-field" and the results are encouraging. As we acquire redundant information on RSSI (each node's RSSI with respect to all the others), future work will focus on exploiting the RSSI among Members to study group behavior, e.g. in terms of proximity, regularity, density, etc. Future work will also involve refinement of our contact detection protocol and comparison with other approaches from the literature. Further, we will also consider adaptive advertising and listening intervals to obtain optimal network performance regardless the number of nodes as well as a context-aware node wake-up strategy to further reduce consumption when a node is alone.



(a) The RSSI from the Member to the Leader (upper trace) and the Member state (lower trace).



(b) Aerial view of the Leader's position recorded with GPS (red dots).

Fig. 6: Results from our case study. The Member awaits the Leader at position 5. The Leader follows the path as indicated by the numbers. Once they meet, the Leader checks in the Member and they continue together.

Acknowledgment

This work has been supported from FBK Smart Community, the CLIMB project and the municipality of Trento.

References

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[2] G. Anastasi, M. Antonelli, A. Bechini, S. Brienza, E. D'Andrea, D. De Guglielmo, P. Ducange, B. Lazzerini, F. Marcelloni, and A. Segatori, "Urban and social sensing for sustainable mobility in smart cities," in *Sustainable Internet and ICT for Sustainability (SustainIT)*. IEEE, 2013, pp. 1–4.

[3] R. Kazhamiakin, A. Marconi, M. Perillo, M. Pistore, G. Valetto, L. Piras, F. Avesani, and N. Perri, "Using gamification to incentivize sustainable urban mobility," in *IEEE Intl. Smart Cities Conference (ISC2)*. IEEE, 2015, pp. 1–6.

[4] M. Razzaque and S. Clarke, "Smart management of next generation bike sharing systems using Internet of Things," in *IEEE Intl. Smart Cities Conference (ISC2)*. IEEE, 2015, pp. 1–8.

[5] M. Cattani, S. Guna, and G. P. Picco, "Group monitoring in mobile wireless sensor networks," in *Intl. Conf. on Distributed Computing in Sensor Systems and Workshops (DCOSS)*, 2011, pp. 1–8.

[6] S. J. Lee, M. C. Kim, Y. S. Kyung, K. K. Jung, J. W. Kim, Y. G. Lee, and K. H. Eom, "The group management system based on wireless sensor network and application on android platform," in *Advanced Computer Science and Information Technology*, 2011, pp. 152–159.

[7] "Lineable," http://lineable.net/, accessed: 2016-04-29.

[8] "CC2650," http://www.ti.com/product/CC2650, accessed 2016-4-20.

[9] M. Bakht, M. Trower, and R. Kravets, "Searchlight: helping mobile devices find their neighbors," in *ACM SOSP Workshop on Networking, Systems, and Applications on Mobile Handhelds*, 2011, pp. 1–6.