CoMon: Cooperative Ambience Monitoring Platform with Continuity and Benefit Awareness

Youngki Lee, Younghyun Ju, Chulhong Min, Seungwoo Kang, Inseok Hwang, Junehwa Song
Computer Science Department, KAIST, Daejeon, Republic of Korea
{youngki, yhju, chulhong, swkang, inseok, junesong}@nclab.kaist.ac.kr

ABSTRACT
Mobile applications that sense continuously, such as location monitoring, are emerging. Despite their usefulness, their adoption in real-world deployment situations has been extremely slow. Many smartphone users are turned away by the drastic battery drain caused by continuous sensing and processing. Also, the extractable contexts from the phone are quite limited due to its position and sensing modalities. In this paper, we propose CoMon, a novel cooperative ambience monitoring platform, which newly addresses the energy problem through opportunistic cooperation among nearby mobile users. To maximize the benefit of cooperation, we develop two key techniques, (1) continuity-aware cooperator detection and (2) benefit-aware negotiation. The former employs heuristics to detect cooperators who will remain in the vicinity for a long period of time, while the latter automatically devises a cooperation plan that provides mutual benefit to cooperators, while considering running applications, available devices, and user policies. Through continuity- and benefit-aware operation, CoMon enables applications to monitor the environment at much lower energy consumption. We implement and deploy a CoMon prototype and show that it provides significant benefit for mobile sensing applications.

Categories and Subject Descriptors
K.8 [**Personal Computing**]: General; C.3 [**Special-Purpose and Application-based Systems**]: Real-time and embedded systems

Keywords
Cooperation, Ambience, Context, Sensing, Energy, CoMon

1. INTRODUCTION
As continuous mobile sensing applications have been increasingly emerging, mobile users are starting to recognize smartphones as a personal sensing platform. Location monitoring services have been popularly deployed, for instance, spatial alarm and trajectory logging services [14][35]. New applications are also appearing; some notify a user about imperceptible information such as UV or dust levels [18], and others provide meaningful statistics from continuous activity observation [22]. These applications provide useful services to mobile users while running in the background, not requiring any explicit user intervention. However, many users are still reluctant to run such applications; they incur significant energy consumption and take up computational resources, potentially disrupting other common uses of the smartphones. This significantly compromises the spread of continuous sensing applications, depriving users of the benefits of running multiple such applications concurrently on their smartphones.

We approach the problem from a novel perspective, by utilizing *in-situ cooperation* of mobile users. We note that, for most people, their daily lives are highly social; they spend a significant portion of their time with others, e.g., family members, friends, colleagues, or even some strangers. According to our study, a user is co-located with acquaintances about 8.5 hours out of 15 active hours of a day, and even more, when accounting for co-location with strangers. In addition, 65% of meetings last for more than 30 minutes and 47% continue for more than an hour, allowing significant opportunities for stable cooperation in continuous sensing. Further, mobile users and their applications often share common interests in many situational contexts related to *ambience* such as locations, atmosphere, and social activities. These contexts can potentially be shared by nearby users, e.g., friends in a social gathering or people traveling in a bus. Thus, users can avoid repetitive sensing and processing redundantly performed by individual users that consume precious energy. This sharing becomes more practical due to the probable cost savings of the sharing. The power consumption for sensing and processing often exceeds the overhead to obtain context data from nearby users; for instance, directly performing location sensing every 10 seconds consumes 410 mW on a Nexus One phone while it consumes only 34 mW to receive the same data indirectly from others through Bluetooth communication (see Section 7 for detailed setup).

As an initial attempt to realize this approach, we propose CoMon, a novel cooperative ambience monitoring platform. CoMon automatically finds cooperators in situ and initiates the cooperation in a way that either enhances its energy capacity or extends its sensing modalities. Applications simply delegate their monitoring requests to CoMon and fully exploit cooperators' resources if available. By employing cooperation, CoMon significantly mitigates the quick battery depletion of devices, or overcomes the absence of specific sensing modalities.

A key challenge in the design of CoMon is how to construct cooperation groups and build network channels for continuous cooperation. We employ a continuity-aware cooperator detection method, which enables CoMon to maintain stable cooperation channels and reduce the complexity in cooperation network management. It leverages heuristic predictors based on social relationships and encounter history, and estimates the potential cooperation duration for candidate cooperators. The system is further designed to support pair-localized cooperation for efficient group management. This makes CoMon more robust against dynamic changes in group membership, by localizing the events such as the departure of a cooperator to a small number of pairs.

Table 1. Context examples and their categories

Context Category		Context Types
Ambience Context	*Spatial Context*	location, ambient sound, place, temperature, humidity, UV, dust-level, noise-level, mood, pollution (CO_2, O_3, …), crowdedness, …
	Social Context	discussion, meeting, conversation, lecture, group exercise, …
Personal Context		activity (walking, standing, …), gesture, health (heartbeat, gait, …), emotion, …

Another important challenge is to provide incentives to cooperating participants. Without benefits, a mobile user would be reluctant to actively participate in cooperation and share her resources for ambience monitoring. However, it is not a straightforward problem to guarantee mutual benefits to all cooperators. Cooperators often run different sets of applications, and possess different sensing devices. Also, they have their own preferences and policies in the use of energy of their devices. Such differences complicate the negotiation to guarantee fair and mutual benefit for cooperators. We propose a benefit-aware negotiation mechanism, which addresses the challenges and builds a mutually beneficial cooperation contract.

CoMon opens a new dimension to address the resource problem for continuous ambience monitoring. Many research efforts have been made to reduce energy consumption for location and context sensing and processing [16][26][27][30], taking an *intra-device optimization* approach, e.g., deactivating GPS based on mobility pattern [30], or optimizing a sound-data processing pipeline applying a noise-level filter [26]. Our *cooperation* approach complements such intra-device optimization techniques, providing further reduction in energy consumption. This additional dimension of benefit is significant, considering continuous and background operation of concurrent mobile sensing applications.

The contributions of this paper are as follows. First, we propose a novel cooperative ambience monitoring platform, CoMon; it significantly improves energy efficiency of smartphones and newly adopts unavailable sensing modalities. Second, we support the practicality of our cooperation approach through motivational studies on ATUS data [1] and Bluetooth-based encounter data [9]. These studies highlight the prevalence of continuous cooperation opportunities as the co-location duration of people comprises a significant portion of a day and social events like meetings continue stably in many cases. Third, as core techniques, we develop continuity-aware cooperator detection and benefit-aware negotiation mechanisms, which enables CoMon to obtain resource benefits from cooperation. Finally, we perform extensive experimental studies based on our prototype implemented over Android phones and custom-designed sensor motes, with diverse sensing capabilities. We show the resource benefits and overheads for diverse cooperation scenarios and applications.

In the rest of the paper, we first motivate CoMon in Section 2 with scenarios and studies on mobility data revealing many opportunities for cooperation. Section 3 describes the model of cooperation benefits and the CoMon architecture. Section 4 and Section 5 describes the two core techniques. We then present our implementation in Section 6 and Section 7 shows experimental results. In Section 8, we discuss other potential issues for CoMon and Section 9 introduces related work. Finally, we conclude the paper in Section 10.

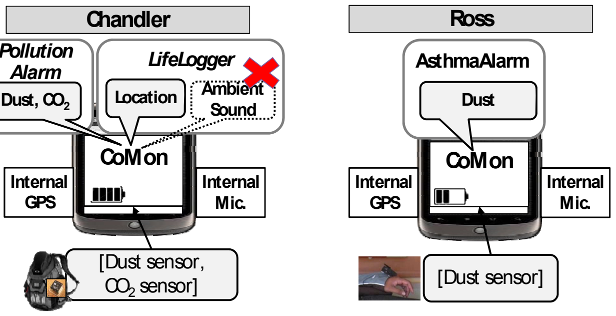


Figure 1. An example cooperation scenario

2. OPPORTUNITY FOR COOPERATIVE CONTEXT MONITORING

Mobile sensing applications have high potential to leverage cooperation between nearby people. As they become popular, many of them will run concurrently, actively utilizing diverse user contexts. Table 1 shows example contexts used by emerging mobile sensing applications [2][18][22][26][27][31][32]. Among them, a number of *ambience contexts* including *spatial* and *social contexts* would be potentially shareable with nearby people.

Along with increasing demands on monitoring diverse contexts, smartphones will incorporate more sensors and people will carry external sensors in diverse form factors, e.g., a watch-type health sensor for everyday precautions against heart attacks, a backpack-attached air-quality sensor to alert the user to possible fine-granule pollutant exposure in everyday life spaces. Considering the globalized nature of consumer product manufacturing, many people will likely have devices with similar sensing capabilities and thus have a high potential for sharing sensing resources.

Understanding that there will be many sharable contexts, two key questions are raised: (1) Does the cooperation result in actual energy benefits for context monitoring? (2) Are there enough cooperation opportunities in the everyday life of mobile users?

We first demonstrate an interesting scenario showing the expected cooperation cases and their benefits in Section 2.1. Note that the energy-related figures used in our scenario are presented based on our actual measurements (See Section 7 for detailed settings including sensor specifications). Then, we show that the opportunities for cooperation are actually prevalent in everyday life through our analysis of human activity and mobility datasets in Section 2.2. The cooperation benefits are further elaborated through comprehensive experiments in Section 7.

2.1 Cooperative Context Monitoring Scenario

Chandler, Ross, and Joey are friends in Manhattan. On Saturday, Chandler plans to meet Ross for shopping in the SoHo area. Chandler always runs two apps, *PollutionAlarm* and *LifeLogger* as in Figure 1. He runs *PollutionAlarm* to avoid exposure to air pollution such as dust and exhaust fumes, and *LifeLogger* to record his route (using GPS) and optionally ambient sound contexts (using the microphone to record music genres, meetings, etc.) [26][27]. Today, he turns off the ambient sound monitoring to extend the phone's battery life. Ross runs *AsthmaAlarm* due to his asthma problem. It monitors the dust levels in the air, a major allergen for asthmatics. While Ross is on his way to SoHo, he discovers that his dust sensor blinks notifying him that there are 'fewer than 3 hours of battery remaining'. Ross gets anxious, regretting that he forgot to recharge the sensor last night.

Table 2. Summary statistics of activity and mobility datasets

Dataset	ATUS	MIT/BT
Data source	American time use study (interview)	Bluetooth scanning trace (period: 5 min)
Participants	13,258	100
Start time	01/01/2010	09/08/2004
Duration	1 year	3 months
# of events	257,193 activities	285,512 encounters

When Ross meets Chandler, Ross's CoMon starts cooperation with Chandler's to monitor the dust level in turn. This reduces the net power-on duration of each sensor by half; the average power consumption by Ross's dust sensor decreases to almost half, from 848 to 487mW, and the estimated sensor lifetime is increased from 3 to 5.2 hours. Note that Ross's smartphone requires a slight additional power expenditure of 25mW to update the dust level to and from Chandler's phone during the cooperation.

Joey was walking in a park near SoHo for his daily exercise; he runs the *CalorieMonitor* application which uses his movement speeds for calorimetry calculation. He also uses *LifeLogger*. On his way home, Joey happens to meet Chandler and Ross and they decide to go to a café. Detecting Joey's devices, Chandler's CoMon system entrusts sound monitoring to Joey's CoMon while supporting location monitoring for Joey instead. This cooperation enables Chandler's *LifeLogger* to again be fully functional by reactivating the disabled sound monitoring. Now Joey's phone turns off energy-intensive GPS sensing which has consumed 440mW; instead, it needs only 11mW to receive location context from Chandler. The additional cost to Joey's device to provide the ambient sound context is marginal (12mW), since he has been monitoring this context for his own purpose. Through the cooperation, the total power consumption of Joey's phone is reduced from 570 to 365mW, increasing its lifetime by about 56%.

Note that CoMon achieves savings on other resource types such as CPU and memory as well. For instance, for ambient sound processing, over 20 operators for feature extraction and context classification such as FFT and GMM run continuously, requiring over 10% CPU cycles on a Nexus One phone. In the above scenario, CoMon allows Chandler to monitor the sound-driven context without such a cost through the cooperation with Joey. Furthermore, the benefit of CoMon is not limited to applications utilizing ambience contexts. Since smartphones are shared by multiple concurrently running applications, the amount of resources saved by cooperation can be leveraged to monitor other personal contexts or for other common uses.

2.2 Study on Cooperation Opportunity

To study cooperation opportunities in the daily life of mobile users, we analyze two public datasets on human activity and mobility behaviors. Table 2 shows their summaries. Note that while there have been many efforts on mobility data analysis, few focused on revealing cooperation opportunities; most of them focuses on revealing human mobility patterns,

ATUS: The American Time Use Survey (ATUS) dataset [1] includes the list of all activities of American participants over a 24 hour period and the acquaintances who were present during each activity. We use the dataset collected in 2010 from 13,258 interviewees over wide age, sex, and occupation distributions. We analyze this data to find the cooperation opportunities in everyday activities, especially in terms of the acquaintances being together.

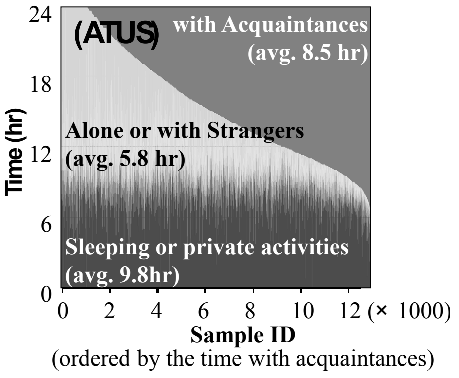


Figure 2. Distribution of the time together

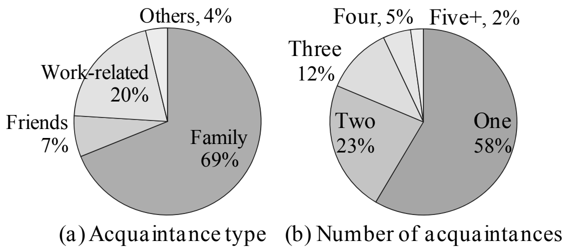


Figure 3. Detailed analysis of the time with acquaintances

MIT/BT: The MIT/BT dataset is the mobility dataset collected from 100 mobile phones of MIT students and staffs [9]. It is collected by Bluetooth scanning performed every 5 minutes. We analyze the encounters between the phones, i.e., the encounters with nearby people including strangers as well as acquaintances. We use the three-month dataset from the fall semester of 2004.

2.2.1 How Many Opportunities for Cooperation?

The longer people are together with others, the more opportunities for cooperative context monitoring we can exploit. To quantify the amount of such time in everyday life, we analyze the ATUS dataset. We do not use the MIT/BT dataset here since it is limited to the devices only discoverable by Bluetooth scanning.

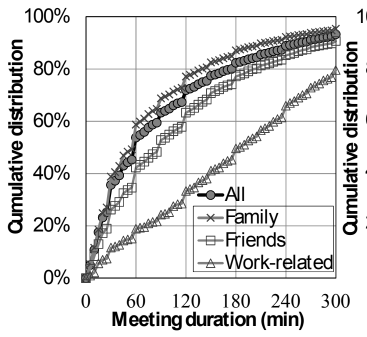
Figure 2 shows the daily amount of time in terms of the presence of acquaintances for every participant. The average time with one or more acquaintances is 8.5 hours. We can confirm that people have lots of cooperation opportunities with acquaintances, i.e., more than one-third of a day. Specifically, 78% of the participants have more than 4 hours of co-located time with others, and 50% have more than 9.3 hours.

Figure 3(a) elaborates on with whom and how long participants spent time with acquaintances, i.e., family (average 5.9 hours), work-related people (1.7 hours), friends (0.6 hours), etc. Figure 3(b) shows the number of acquaintances a user is together with; it gives an intuition on the number of cooperator candidates at a time. For 42% of the time, people are with more than one acquaintance, giving more chances of cooperation.

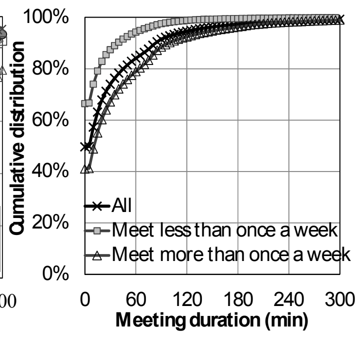
Note that the opportunities for cooperation are not limited to those with acquaintances but can include those with strangers, e.g., a user in a bus can cooperate to monitor the route of the bus with other passengers. However, the ATUS dataset does not contain encounters with strangers.

2.2.2 Continuity of Cooperative Monitoring

We study the continuity of meetings, i.e. how long people are together during an encounter. Once cooperative monitoring has been



(a) ATUS



(b) MIT/BT

Figure 4. CDFs for meeting duration between people/devices

established when they are together, this monitoring could continue as long as they remain together. Long-lasting cooperation would enable prolonged support for applications.

We study the ATUS dataset to identify how long the cooperation with acquaintances could last. Figure 4(a) shows the distribution of meeting durations. We define a meeting as consecutive activities with the same people, e.g., shopping at a department store with the spouse, then travelling together to a restaurant and having dinner. The figure shows the distributions for different acquaintance types. For 'all', it shows that 65% of meetings with acquaintances last for more than 30 minutes and 47% last for more than one hour. From these results, we can obtain typical durations of cooperative monitoring between acquaintances.

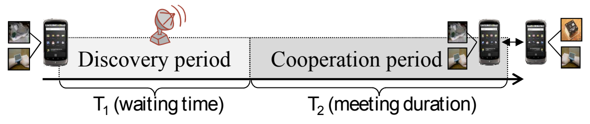
Figure 4(a) also shows that the meeting duration largely depends on the type of acquaintance; meetings with work-related people usually continue for the longest time, followed by friends and family. When there are multiple acquaintances nearby, we can consider their social relationship to select better promising cooperators in terms of continuity.

We study the MIT/BT data to determine the duration of meetings between two arbitrary mobile users without distinguishing between strangers and acquaintances. Figure 4(b) shows the distribution of encounter durations. We define the duration of an encounter as the time during which a user's mobile device is consecutively scanned by the other users'. Overall, the encounter duration is shorter than that in ATUS; 50% of people pass by within a duration of 5 minutes (a single detection only) and 26% stay together for more than 30 minutes. This is mainly because it includes strangers and people passing by. Also, if two people temporarily move apart from each other for a few minutes, e.g., one goes to the restroom, the encounter is segmented and treated as two separate meetings. Importantly, we found that the meetings between people who meet frequently, namely familiars, last longer. For people who meet once a week or more, 36% of the meetings between them last for more than 30 minutes while only 13% of the meetings with the others do. This implies that meetings with familiars would yield longer cooperative monitoring opportunities.

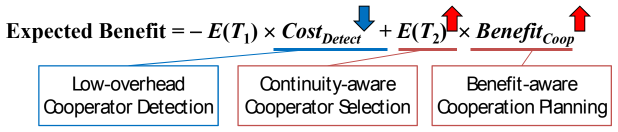
3. COMON DESIGN

3.1 Benefit-aware Cooperation Approach

A key goal of CoMon is to maximize the energy benefit from the opportunistic collaboration with nearby users. For effective design of CoMon, we first attempt to model the energy benefits obtainable from the cooperation with a cooperator, as shown in Figure 5. In the model, we divide the operation time into two periods, i.e., discovery period and cooperation period.



(a) Cooperation model for a cooperator



Expected Benefit = – E(T1) × CostDetect + E(T2) × BenefitCoop
Low-overhead Cooperator Detection
Continuity-aware Cooperator Selection
Benefit-aware Cooperation Planning

(b) Cooperation benefit and approaches

Figure 5. Cooperation benefit model

In the discovery period, the system attempts to detect nearby cooperator candidates. This incurs cost, which can be represented as $Cost_{Detect} \times E(T_1)$; $Cost_{Detect}$ is the average discovery cost per unit time, T_1 is the random variable of the waiting time until meeting a cooperator, and $E(T_1)$ is the expected waiting time. Once the cooperation starts with a cooperator, it will produce benefit. We model the benefit for the cooperation period as $Benefit_{Coop} \times E(T_2)$, where $Benefit_{Coop}$ is the average benefits from cooperation per unit time and $E(T_2)$ is the expected duration of cooperation. Taking all the cost and the benefits into account, the expected total benefit can be specified as follows:

$$Expected\ Benefit = Benefit_{Coop} \times E(T_2)\ -\ Cost_{Detect} \times E(T_1)$$

To increase the expected energy benefit, we devise the steps of cooperation as below.

Cooperator detection: The first step is to continuously detect nearby users as potential cooperators. The discovery period should be carefully chosen; a long interval reduces the discovery cost, i.e., $Cost_{Detect}$, but might decrease the potential cooperation duration, $E(T_2)$. (See Section 4)

Cooperator selection: Increasing the cooperation period T_2 is crucial for higher benefit $E(T_2)$. We explore the potential to predict the meeting durations upon discovery of a cooperator candidate. The negotiation for real cooperation starts only with the candidates who can lead to long enough cooperation to provide benefits. We develop the continuity-aware cooperator selection method (Section 4.1).

Cooperation planning: To increase the benefit per unit time, it is important to carefully determine a cooperation plan, e.g. selection of contexts to share or distribution of tasks to different devices. Different plans could significantly influence the benefit from the cooperation. We develop a planning method, which carefully decides the cooperation plan for higher $Benefit_{Coop}$. It ensures mutual benefits to both cooperators, since either of them would be reluctant to participate if benefits are not mutual (Section 5).

3.2 Architecture Overview

We carefully designed the architecture of CoMon as shown in Figure 6 applying the benefit-aware cooperation approach. It runs as a middleware on top of a smartphone OS and additionally supports external sensing devices [17][21]. CoMon provides mobile sensing applications with intuitive APIs, allowing the applications to specify the contexts of interest (e.g. location, activity) in a declarative query [16]. Consider a pollution monitor application that wants to monitor CO_2 level with 90% of accuracy every 30 seconds. Then, it specifies the query as follows:

CONTEXT CO_2 level **ACCURACY** 90%
PERIOD 30 Seconds **DURATION** Always

CoMon processes registered queries by leveraging cooperation opportunities with nearby users; it handles the query with the user's own devices if it finds no potentially beneficial cooperator. CoMon takes charge of all the underlying tasks for opportunistic cooperation, which are transparent to the applications. In terms of applications, the quality of service (QoS) provided by CoMon might vary from time to time due to the heterogeneity of devices or dynamic system situations. We believe that the slight QoS difference caused by cooperation does not cause severe problem for many applications; current Android sensing APIs do not also guarantee fine-granule QoS for GPS and accelerometer sensing. For the applications that have hard QoS requirements, CoMon may not initiate cooperation or can check QoS condition further while planning.

The benefit-aware cooperation approach is realized by two key components: *cooperator detector* and *cooperation planner*. The *cooperator detector* dynamically discovers nearby devices by periodic Bluetooth scans at a small overhead, and selects candidates that will potentially stay in the vicinity for a long period (long stayers) (Section 4). The *cooperation planner* negotiates with the selected one, and then decides the best cooperation plan (Section 5). Note that when the cooperator leaves, CoMon instantly processes the contexts provided by the cooperator with its own devices.

According to the cooperation plan, the *context processors* on the smartphone and sensors continuously process the requests and deliver the processing results to the applications and cooperators (via Bluetooth in current implementation). It incorporates a variety of processing modules for sensing, feature extraction, and context classification to support diverse types of contexts.

The *device manager* provides the cooperation planner with up-to-date energy information, required to make a proper plan. As a basic support of privacy, CoMon employs *access controllers*, which restrict unauthorized accesses to certain contexts. CoMon allows users to specify the access rules about what context information can be shared with whom. We further discuss the privacy issues and access rules in Section 8.

We employ a smartphone-centered architecture; external sensor devices and their data are exposed to cooperators only through smartphones. We suppose that external sensors would be designed in a small size with minimal computational resources. It would be hard for such sensors to perform as an independent participant for cooperation since they are limited in supporting multi-user connections or processing raw sensor data with complex modules.

3.3 Design Considerations and Choices

We below present key considerations and our choices for the design of practically working cooperation systems like CoMon.

Long-term cooperation: Dynamic changes of cooperators due to their mobility could incur overheads for frequent discovery, negotiation, and connection management. To minimize such overheads, CoMon targets the cooperation with long stayers only. Even when a user walks around in crowded Manhattan, CoMon selects the cooperators only among acquaintances doing the activity together, or familiar strangers who stay together for more than a certain amount of time. We find that even with only long-term cooperation, there are sufficient opportunities.

Pair-wise negotiation: When there are multiple cooperator candidates, it is important to determine how to organize the group for cooperation planning and execution. Our key idea is to localize

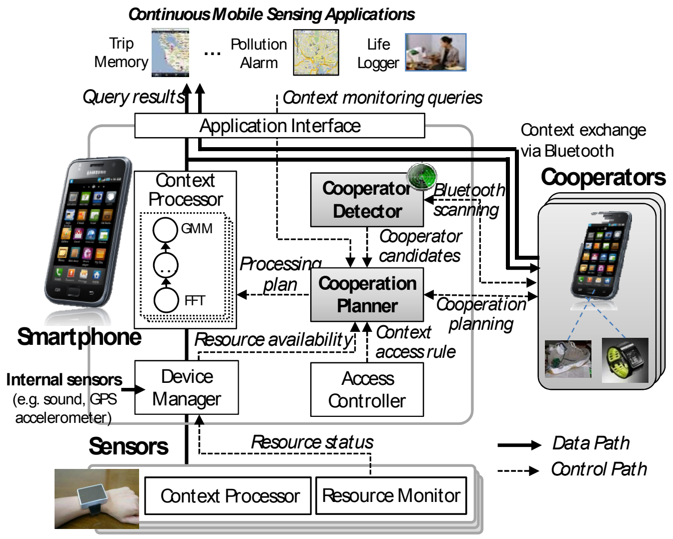


Figure 6. CoMon architecture

the effect of membership changes. CoMon performs the cooperation in the unit of a pair to localize the effect within some pairs. It negotiates with the cooperator candidates one at a time and incrementally continues the negotiation. An alternative approach would consider the whole group as a single cooperation unit, and perform a group-wide negotiation at once. Although this approach would lead to the group-wide resource optimum, it is relatively vulnerable to the mobility of users. Whenever a single cooperator joins or leaves, all the remaining cooperators should re-perform the negotiation process. Also, the group-wide negotiation significantly escalates the complexity of cooperation planning due to numerous options and policy conflicts.

Context-level service as cooperation interface: For negotiation, an important design choice is the appropriate abstraction level in exposing a mobile user's resources to cooperators. CoMon exposes underlying resources of the user's devices as context-level services. A context hereby means high-level information extracted by classifying or aggregating raw sensing data, e.g., a place category extracted from ambience sound or a 30-second averaged dust count in the air. The context-level service hides heterogeneity and dynamics of other cooperators' resources. Also, context-level exchanges could greatly save energy which might be high if high-rate raw data are exchanged. We assume that there would be consensus on a common context model as in [30], which could help extend the scope of the cooperation. Based on such model, different applications running over heterogeneous devices can share and exchange context information. Even with the common consensus on a context, different applications may require different level of accuracies and sampling and processing intervals. CoMon can evaluate such condition in the planning process but we do not handle such cases here for simplicity.

4. COOPERATOR DETECTION

A first step of CoMon operation is to find out the cooperators who will provide benefits through cooperation. The benefits will be provided when 1) cooperators stay longer enough to breakeven the overhead to start cooperation, and 2) there exists sharable contexts for which local execution takes more resources. A good detection method should locate such cooperators with low overhead. However, this is a significant challenge. First, it is difficult to accurately

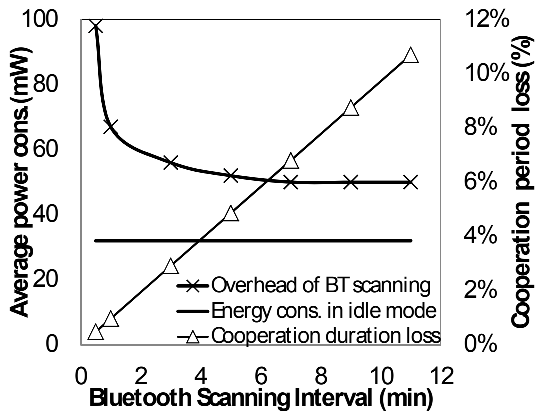


Figure 7. Tradeoff between energy overhead and potential loss in cooperation time

predict the length of encounters in advance. Moreover, it is costly to check with all candidates if they have been monitoring contexts that are sharable and beneficial.

4.1 Bluetooth-based Cooperator Detection

An early decision to build CoMon lies in selecting networking interfaces for power-efficient communication among nearby users. As candidates, we consider several popularly used interfaces such as ZigBee, Bluetooth, and WiFi-Adhoc. ZigBee achieves low power but at low bandwidth. WiFi-Adhoc makes opposite tradeoffs and Bluetooth is in the middle. Also, each protocol has its own characteristics, such as broadcast supportability and limited number of connections. The current CoMon implementation adopts Bluetooth, mainly because the interface is pervasive in most commodity smartphones. Also, it provides reasonable coverage, i.e., about 10 meters, for context sharing among the users who do the same activities. Note that for context sharing, protocols like ZigBee might be more adequate once available; it consumes an order of magnitude less power than Bluetooth with reasonable bandwidth (<250kbps).

To detect cooperator candidates, CoMon performs periodic Bluetooth scans, a common method for identifying devices in proximity [23]. A key issue for the discovery lies in deciding an appropriate scan interval T. More frequent scans incur higher energy overheads but enables quicker discovery of potential cooperators. The probable loss in cooperation duration would be $0.5T$ on average. We study the potential tradeoff by measuring power consumption for the scanning on a Nexus One phone, and estimating the potential loss in cooperation duration from ATUS dataset. Specifically, the loss is estimated by subtracting $0.5T$ from all the activity duration with acquaintances, assuming the acquaintances as potential cooperators. Figure 7 shows the results. A good value of T can be carefully determined leveraging the results. For instance, CoMon can use 5-minute interval, where the power consumption starts to saturate. At this interval, the average loss in potential cooperation duration would not be significant, i.e., 4.8% out of expected cooperation time per person; note that many acquaintances are likely to stay for a long period of time.

In addition to the energy overhead, the available network bandwidth is reduced during scanning [12]. It may make the connections with cooperators unstable, even disconnected. Against such instability, CoMon buffers the messages and defers the exchange during the scanning for 10 seconds. Also, CoMon stops scanning when the current cooperation benefit exceeds a certain threshold.

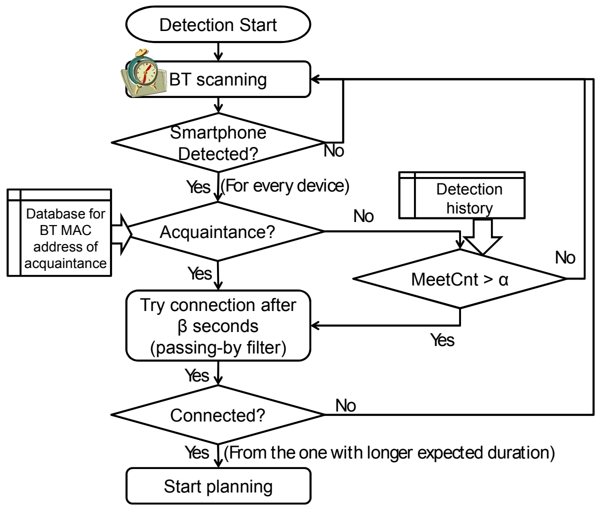


Figure 8. Continuity-aware cooperator selection

4.2 Continuity-aware Cooperator Selection

CoMon needs to distinguish cooperator candidates that are likely to stay together among the ones discovered. There would be a broad spectrum of candidates in terms of potential meeting durations, from those quickly passing by to those staying for hours. It would clearly be useful to know the potential meeting durations with the candidates.

A most interesting observation to automate estimation was made with respect to the social relationship between the potential collaborators. From our study on the ATUS dataset, meetings with acquainted people, e.g., family members, friends, and co-workers, are likely to continue long enough; this can be expected since the acquainted people usually share common purposes for meetings, e.g., doing an activity together. Also, the expected meeting durations differ according to the types of acquaintances or their activities, e.g., socializing or working.

For unknown relationships, the system can further utilize previous encounter or meeting histories. One may expect that the duration of a meeting with a person can be estimated from the distribution of the previous meeting durations with that person. However, our studies on MIT/BT dataset saw little correlation between the durations of previous meetings and the current one. Interestingly instead, we could see that number of previous encounters affects the current meeting duration.

Based on the observation, we develop a *continuity-aware cooperator selection* method. It distinguishes the candidates into multiple categories with different expected cooperation duration, e.g., passing-by, acquaintances, and strangers with multiple encounters. CoMon preferentially starts to negotiate with those in the category of longer expected duration. Figure 8 shows the overall flow of the selection process.

Acquaintance selection: Following the observation, the method first selects the devices owned by acquaintances. According to the ATUS data, the average expected continuation time is 51 minutes for acquaintances. The method further utilizes the type of acquaintances if the information is known, e.g., the expected cooperation time with a friend can be estimated as 128 minutes (see

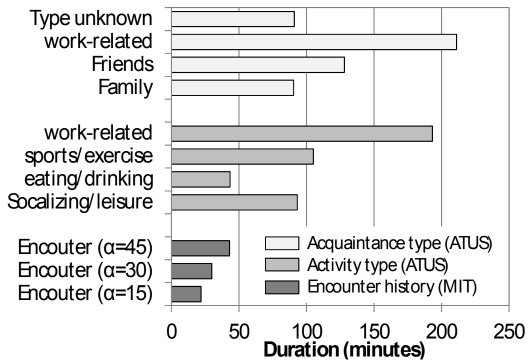


Figure 9. Expected cooperation duration

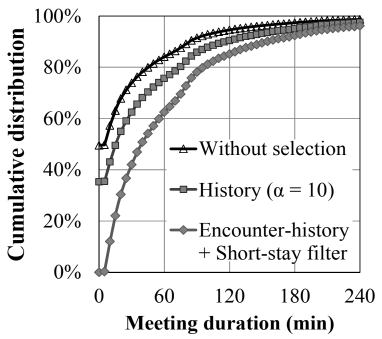


Figure 10. The effect of the cooperator selection (MIT/BT)

Figure 9). It can further consider activity types once it becomes available to the smartphones. To identify if a device is owned by acquaintances, the detector maintains a small database to match acquaintances' devices with their Bluetooth MAC addresses. To obtain this information, CoMon asks the user to manually designate the owner of a device from the phone book upon the first pairing; CoMon allows the user to disable the new pairing to avoid frequent new pairings with strangers.

Encounter history-based selection: When the cooperation with acquaintances are not possible, however, the method further finds other opportunities with the devices that have been frequently discovered more than a certain threshold, α within a time period, e.g., a month. Even when they are not in the contact list, they are highly likely to be a familiar stranger [28], e.g., a person in the same commuter bus. Figure 9 also shows the expected cooperation duration for different α values estimated from MIT/BT dataset.

Short-stay filter: Lastly, the method applies a *short-stay filter* to exclude passing-by devices out of the devices recommended by the two predictors. This is important since not every encounter by acquaintances or frequently meeting people leads to a longer meeting. The key idea is that the devices that already stayed for a certain time are likely to stay longer. We can find an analogy with a least-recently-used (LRU) paging algorithm in OSs. The method attempts to connect to the devices selected by predictors after a certain time, β seconds, not instantly after the discovery.

We validate the effectiveness of the encounter history-based selection and short-stay filter by emulating them over the MIT/BT dataset (α=10 in a two-week window, β=300 seconds). We skip acquaintance selection as we do not know about the relationships among the participants with the dataset. Figure 10 shows our method effectively selects the users who are likely to stay longer. For example, whereas the probability that a meeting lasts over 30 minutes is 26%, it increases to 38% when the encounter-history selection is applied, and further increases to 58% with the short-stay filter. The short-stay filter seems to be quite useful while the effect of encounter-history selection is not very significant. We thus apply the encounter-history based predictor only when cooperation is not available with acquaintances. We acknowledge that a large-scale deployment will be more realistic than our emulation with the MIT/BT dataset, and are working towards that.

5. COOPERATION PLANNING

With the candidates recommended by the detector, CoMon conducts cooperation planning, to decide which contexts to share and trade. For each user, the main goal of the planning is to maximize her benefit. For the system, the goal is to provide mutual and fair benefits for cooperators.

However, providing such maximized and mutual benefits is not a simple problem. An initial solution is that cooperators take turns to monitor the common contexts when the cooperation leads to benefits. For this, the system just needs to identify common requests of cooperators. For each context, it compares the energy demands to process it locally with the potential demands upon cooperation. Note that the cooperation is often beneficial for the contexts requiring energy-intensive sensing like GPS or dust sensing, or heavy processing over high-rate data like ambient sound context. For the beneficial contexts, a cooperator could monitor them for certain time duration, e.g., ten minutes and then the other monitors them for the next ten minutes.

Such a solution works in simple cases, but it needs to be further improved to deal with complex system environments. A key challenge results from the complexity in benefit estimation. The cooperation benefit cannot be statically determined in advance; even for the same cooperation pattern, the benefit could vary depending on resource availability, operational applications, and user policies. To be specific, first, the energy demand to monitor a context might be differently accounted considering other concurrently monitored contexts. CoMon processes multiple contexts in a shared way; it figures out the overlapping tasks among contexts, e.g., sensing, processing and communication tasks, and eliminates redundancy. Accordingly, the cooperation benefit for a context needs be evaluated, taking such shared evaluation into account. Moreover, cooperators expect different types of benefits and have different policies on energy use. A user who will be outside quite a while would want to save energy as much as possible, but the one who will soon go home would not mind consuming energy if he can benefit from new contexts.

The planning becomes even more challenging as we target future environments where a user carries a number of wearable sensors together with a smartphone. In this case, monitor-able contexts among users vary quite much, and sharing common contexts only provides limited benefits. Also, the user policies could be more complex reflecting the in-situ availability of sensing devices and their remaining energy in combination. For example, in Section 2, Joey obtains significant benefit from location sharing, saving battery of his power-hungry smartphone. However, the benefit would be relatively less if he uses a powerful external GPS; in this case, sharing through the smartphone might be even a loss.

5.1 Cooperation Planning Problem

To understand the problem in depth, we first clarify the problem and cooperation benefit. According to our context-level sharing principle, we describe a cooperator, *u*, as follows:

```
Input: $\{ctx_d\}$, a set of contexts to monitor
Output: cost to execute $\{ctx_d\}$
1. EDVector ← GetEDVector($\{ctx_d\}$)
2. totalEC ← 0    // init total energy consumption
3. for ∀ $d_j$ where $d_j$ is a device in EDVector
        totalEC ← totalEC + $weight_j$ · EDVector($d_j$)
4. Return totalEC
```

Figure 11. A cost function for a policy 2

Def 1. A cooperator, *u*, is specified as: ***u* = <*D*, *S*, *P*>**, where

- *D* is a set of demanding contexts, $\{ctx_d\}$, by applications; the set is obtained from the registered queries in CoMon.
- *S* is a set of supply-able contexts, $\{ctx_s\}$, which CoMon can monitor and provide to cooperators; CoMon identifies the set based on available devices and their resource availability.
- *P* is a policy that denotes the desirable benefit from the cooperation. It is given by the user based on his preference or resource situation. The policy is substantialized as a cost function, $cost_P$, within the system. If $cost_P$ is reduced as a result of cooperation, the cooperation is considered beneficial.

Now, given two cooperators u_1=<D_1, S_1, P_1>, and u_2=<D_2, S_2, P_2>, the cooperation planning problem is to find *a cooperation plan*, *CP*, as its output for the estimated cooperation duration, where

- CP = {(ctx_c, u_i, t) | ctx_c is a context to cooperatively monitor,
  u_i is a cooperator in charge, either u_1 or u_2,
  t is a time duration to take charge},

such that $cost_{p1}$ and $cost_{p2}$ should decrease by applying CP. In each person's viewpoint, the purpose is to minimize her cost function but it could conflict with the purpose of the cooperator.

5.2 Cooperation Benefits and Policies

A user can apply diverse policies to describe his preferential benefits from cooperation. We first introduce useful example policies which are described in terms of device resources and application supportability.

Policy 1: A basic policy is to save the battery consumption of a smartphone for context monitoring. Since a phone is a generic personal computing platform utilized for diverse applications, it is better to save the battery power by default.

Policy 2: When a smartphone works together with external sensing devices, a user might want to consider the battery status of other devices as well. According to the importance and use cases of devices, a policy can be defined to reduce the weighted sum of power consumption over distributed sensing devices.

Policy 3: In terms of application supportability, a policy can be defined as to increase the number of supported queries. CoMon may not continue to support some requests due to shutdown or low battery level of corresponding devices. The policy attempts to resume the support for such requests through cooperation.

Policy 4: Sometimes, it is expected that a user will recharge the devices after a certain time, *T*, e.g., 3 hours. In this case, a policy is specified to increase the running time up to 3 hours for all applications if some cannot be satisfied only with local resources.

CoMon provides several system functions to enforce policies as cost functions. The key primitives are *getEDVector*({*ctx*}) and *getEAVector*(). *getEAVector()* returns the remaining energy of all sensing devices. Given a set of contexts to monitor, {*ctx*}, *getEDVector*({*ctx*}) returns the expected power consumptions on

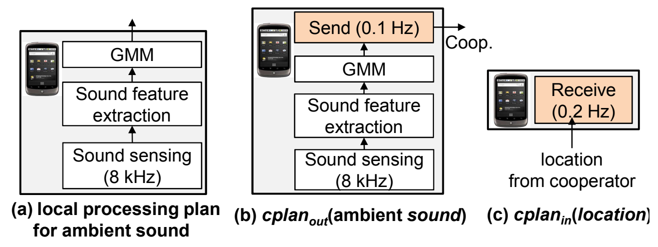


Figure 12. (a) a local plan for ambient sound context (simplified), (b)(c) *cplans* for *case_ex*(*sound*, *location*) when Joey cooperates with Chandler in the scenario in Section 2.

relevant sensing devices. For example, *getEDVector*({*Dust*}) returns an energy demand vector, (28.5*mW,* 720.7*mW*), where the elements represent the energy demands on the smartphone and dust sensor, respectively. Figure 11 shows an example cost function, $Cost_{P2}$, realizing policy 2 based on the primitives.

To compute the energy demands, CoMon manages energy use profiles for the sensing, processing and communication tasks required to monitor contexts; currently, energy use are profiled offline while on-line profiling can be applied further. In CoMon, the processing of a context is represented as a graph of tasks, denoted as a *processing plan*. Figure 12(a) shows the example plan for the ambient sound context (see [15] for a detailed graph). CoMon estimates the energy demand to execute a plan by adding the energy demands for all tasks constituting the plan. Note that CoMon properly reflects the effect of the shared processing; the energy demands for redundant tasks between multiple contexts are accounted only once.

We build the system functions extending our previous systems [17][21]; they leverage such information for the coordination of multiple applications' resource use over personal devices. While they utilize multiple alternative plans for a context, we suppose that CoMon has a single plan for each context to focus on cooperation.

5.3 Benefit-aware Negotiation Mechanism

Careful cooperation planning is essential to provide cooperation benefits under complications in running applications, available sensing devices and their energy, and user policies. It becomes more complicated to provide mutual and fair benefits for both participants as they may have conflicts in planning decisions.

To address such challenges, we develop a benefit-aware negotiation mechanism. As a key idea, the mechanism pursues the fairness of opportunity to make beneficial cooperation decisions by themselves, rather than guaranteeing mutually identical benefits to cooperators; the identical benefit is not even possible due to participants' different policies and energy availability. In this principle, the mechanism utilizes one-to-one context exchange as a first-stage negotiation unit, providing each cooperator a chance to weigh up the unit by its own cost function. For each unit, it estimates the benefit reflecting in-situ resource availability and concurrent requests. Then, the benefit is cross-validated by each cooperator to ensure mutual benefits; the cases beneficial to only one side are excluded in advance, so that the planning results ensure the mutual benefit. Finally, the mechanism allows the cooperators to take turns to select the unit of exchange, providing each participant with fair opportunities to maximize its benefit.

In more detail, the mechanism introduces a *cooperation case*, as an atomic unit of cooperation planning; we describe the mechanism in perspective of a cooperator, u_1. We identify two representative types of cooperation cases as follows. Note that cooperation cases are built on a context level, hiding the low-level resource details of a cooperator.

- **Exchange of two contexts, *ctx_{out}* and *ctx_{in}***, denoted as ***case_ex(ctx_{out}, ctx_{in})***, is a case that u_1 obtains a context ctx_{in} from u_2 in exchange of providing ctx_{out}. This case enables the participants to save the energy by delegating the costly monitoring of a context or obtain an unavailable context.
- **Co-monitoring of a context, *ctx_{co}*, *case_co(ctx_{co})***, is a case that u_1 and u_2 monitor ctx_{co} in rotation. This case enables the participants to save the energy by halving the monitoring duration of the context.

With the cooperation cases, our planning method is performed in following three steps.

Step 1. Cooperation case generation: First, participants generate applicable cooperation cases by exchanging their demanding and supply-able contexts, i.e., *D* and *S*, with each other. The generated cases include a set of exchange cases, *EX*, and a set of co-monitoring cases, *CM*, where

- $EX = \{case\_ex(ctx_{out}, ctx_{in}) \mid ctx_{out} \in (S_1 \cap D_2),$
  $ctx_{in} \in (D_1 \cap S_2), ctx_{out} \neq ctx_{in}\}$, and
- $CM = \{case\_co(ctx_{co}) \mid ctx_{co} \in (S_1 \cap D_1 \cap S_2 \cap D_2)\}$.

For an exchange case, ctx_{out} is the one that u_1 provides and u_2 demands. ctx_{in} is vice versa. Second, a co-monitoring case is generated for a context that u_1 and u_2 both can provide and demand at the same time. If a cooperator has been already cooperating with another one u_3, it excludes the contexts involved in the cooperation with u_3 from its *S* and *D* for the case generation, following our pair-localized negotiation design.

Step 2. In-situ benefit estimation and cross-validation: The second step is to estimate the benefit of each generated cooperation case and exclude the cases that provide only one-side benefit. Since the benefit of a case can be differently estimated depending on each participant's policy and energy availability, the benefit estimation is separately done by each participant based on its *cost* function. Based on the estimated benefit, each participant excludes the cases that are not beneficial to the participant. Then, they exchange the list of the cases to exclude the cases that are not beneficial to the other participant as well. The cross-validation results in only mutually beneficial cases.

Details on benefit estimation. The key of this step is to estimate the benefits for a cooperation case. In detail, the benefit is calculated in two sub-steps: 1) introducing *a cooperation plan*, and 2) policy-based benefit calculation applying the new plan.

Internally, a cooperation case introduces a new processing plan to monitor the corresponding context. We denote such newly introduced plan as *a cooperation plan*, *cplan*, while denoting the original local plan as *lplan*. The new cooperation plans are differently introduced for exchange cases and co-monitoring cases. For an exchange case, *case_ex*(ctx_{out}, ctx_{in}), a new $cplan_{in}(ctx_{in})$ is created for ctx_{in}. The $cplan_{in}(ctx_{in})$ simply consists of a task to receive the results for ctx_{in} from the cooperator. For ctx_{out}, a new $cplan_{out}(ctx_{out})$ is built by inserting a task to send results at the end of its original local processing plan. Figures 12(b) and (c) show example *cplans* created by the *case_ex*(*sound*, *location*) for Joey in Section 2.1. For a co-monitoring case, *case_co*(ctx_{co}), a cooperation plan $cplan_{in}(ctx_{co})$ is used for every first half of rotation epoch to

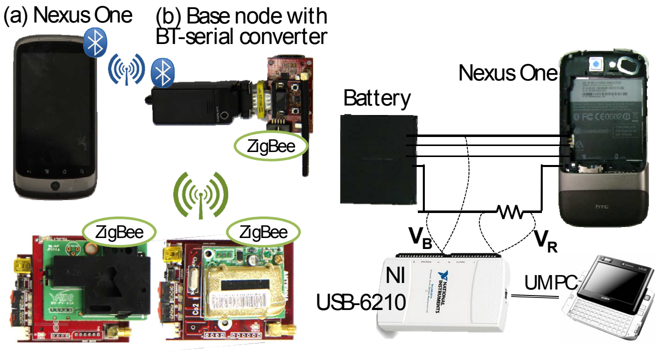


Figure 13. Hardware and energy measurement setup

receive ctx_{co} and $cplan_{out}(ctx_{co})$ is used for the second half to provide ctx_{co}.

With the new *cplans*, CoMon recalculates the cost using the *GetEDVector*() and *GetEAVector()* function. Then, the benefit is calculated by subtracting the new cost from the previous cost before applying the *cplans*, i.e., only with local plans.

Step 3. Turn-by-turn case selection: The final step is to select the validated cooperation cases one-by-one in turn. A participant who has a turn selects the case of the maximum estimated benefit and notifies it to the other. After the selection, the participants delete the cases associated with the contexts in the selected case. For example, if a participant selects the co-monitoring case of location context, both participants delete the cases that exchange the location context with another context. The selection process continues until there is no case to select. Such turn-based selection provides each cooperator with fair opportunity to maximize its benefit. After the selection, the cooperation planner applies and executes the cooperation plan for the selected cases.

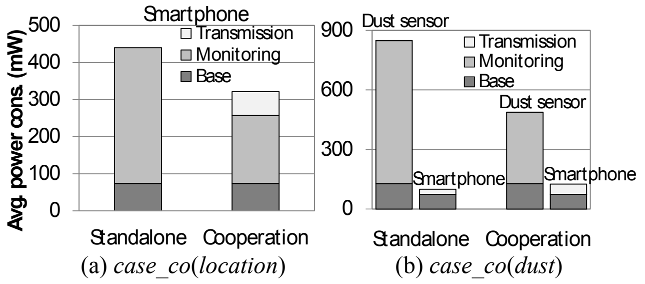
6. IMPLEMENTATION

We prototyped CoMon on Android phones and various types of sensor devices. Figure 13 shows our hardware setup. We used Google Nexus One with 1GHz CPU, 512MB RAM. We connect a base sensor node to Nexus One via Bluetooth-to-serial converter to support ZigBee communication between Nexus One and sensor devices. We used commercially available ZigbeX sensor motes running TinyOS 1.1.11. They are equipped with Atmega 128L MCU, CC2420 RF transceiver supporting ZigBee protocols, and an additional extension board of dust and CO_2 sensors.

We developed mobile-side CoMon architecture as a background service on the Android platform. It provides applications with a service interface for query registration. For the registered queries, CoMon continuously delivers context monitoring results via a callback message handler. The major components implement the Android handler interface and interact with each other through message exchanges. On the sensors, we implemented the sensor-side architecture in NesC. For efficient utilization of limited sensor resources, the architecture identifies the common tasks in multiple application queries, and utilizes shared buffers to avoid duplicate data sensing and reduce the transmission by packing messages for different requests into a single packet.

7. EXPERIMENT

To demonstrate the effectiveness of CoMon, we evaluate the system based on the prototype described in Section 6. First, we present the



(a) *case_co*(*location*) (b) *case_co*(*dust*)

Figure 14. Power consumptions for co-monitoring case

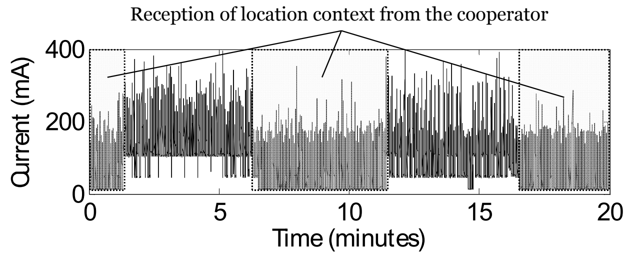
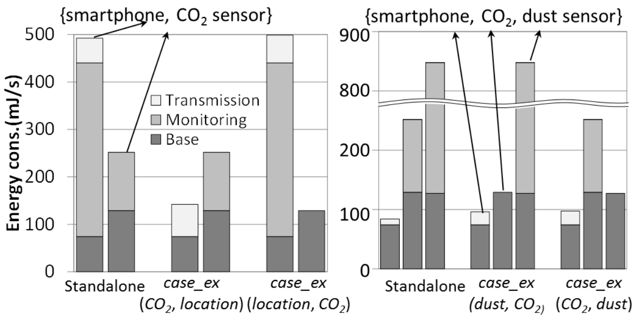


Figure 15. Current draw behavior of *case_co*(*location*)



(a) {*location*, CO_2} (b) {*dust*, CO_2}

Figure 16. Power consumptions for exchange cases

energy benefit achieved in diverse cooperation cases. Second, we show that our cooperation planning method effectively provides mutual benefit. Third, we investigate the energy overhead for the cooperation. Lastly, we deploy CoMon on 12 people's devices and present interesting results and valuable lessons. For the power measurements besides the deployment experiments, we used a data acquisition tool, NI USB-6210, as shown in Figure 13(e). The tool measures the voltage (V_B) of the battery and the voltage (V_R) across a register at a sampling rate of 125 kHz. We measure the energy consumption over a certain period of time, 10 minutes by default, as follows:

$$Energy = \sum Power \times Time = \sum_t \left(\frac{V_R}{R} \times V_B\right) \times (t_i - t_{i-1})$$

For each sample, the power is obtained by multiplying V_B by the current that is calculated by dividing V_R by the resistance value, R (in our setting, 100mΩ). As a metric for energy, we use the average power consumption (mW), which is obtained by dividing the total energy use for the whole time period by the time.

7.1 Energy Benefits of Cooperation

We evaluate the energy benefits achieved by cooperative context monitoring. We measure and analyze the energy saving of the smartphone and the sensor devices for basic cooperation cases, i.e., co-monitoring cases and exchange cases. We compare the power consumption after applying the cooperation cases against the non-cooperative, standalone setting.

For the detailed analysis, we break down the power consumption into three states; *base*, *monitoring*, and *communication*. The *base* represents the power consumption for the primitive operations of the smartphones and the sensor devices. The *monitoring* includes the power consumed by data sensing and processing. The *communication* is the energy to send and receive the data. This includes both cases to communicate with a person's own external sensor devices and the cooperators; note that both communications are done via Bluetooth.

Co-monitoring cases: Figures 14(a) and (b) show the average power consumption for two co-monitoring cases, *case_co*(*location*) and *case_co*(*dust*), respectively. Figure 14(a) shows that CoMon achieves 27% of power saving on the smartphone through the co-monitoring of the location context, i.e., from 440 to 321mW. We expect that the energy saving extends the lifetime of the smartphone by 37%. The major contribution comes from halving the total duration of GPS activation. Note that the amount of power saving is less than the exact half, because of the 'base' consumption and the Bluetooth transmission for context exchange. Figure 15 shows the current drawn by the smartphone for 20 minutes for *case_co*(*location*). It clearly shows the periodic activation of the GPS, and the lower current draw when the location context is being provided by the cooperator.

Figure 14(b) shows the results for *case_co*(*dust*), which employs an external dust sensor. CoMon reduces the power consumption of the dust sensor by 43% (from 848 to 487mW), since it is turned off for a half of the monitoring duration. On the other hand, the power consumption of the smartphone slightly increases by 26mW as it transmits the monitoring results to the cooperator during its monitoring turn. This overhead is marginal in most cases; this is because, even in the standalone setting, the smartphone consumes the energy for the '*transmission'*, to receive the data from the external sensor device. Taking such overheads or not is governed by the user's policy.

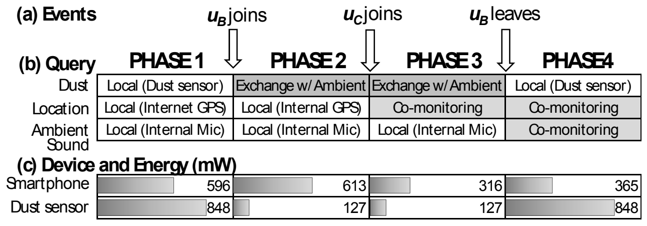
Exchange cases: Figure 16(a) shows the average power consumption for two exchange cases: when the user takes charge of CO_2 in return of location (*case_ex*(CO_2, *location*)), and vice versa (*case_ex*(*location*, CO_2)). For *case_ex*(CO_2, *location*), CoMon significantly reduces the power consumption of the smartphone (492 to 142mW) by deactivating its GPS; the additional cost to deliver its CO_2 context is insignificant, i.e., 7mW. The consumption of the CO_2 sensor remains the same at 251mW. In contrast, for *case_ex*(*location*, CO_2), the power consumption of CO_2 sensor is largely reduced from 251 to 129mW, whereas the smartphone slightly consumes 9mW of more power to transmit the location context. Figure 16(b) shows the exchange cases of CO_2 and *dust* contexts. These cases provide energy benefits similarly as shown in Figure 16(a).

7.2 Cooperation Planning for Mutual Benefit

We validate our cooperation planning mechanism and its effectiveness in terms of mutual benefits. We conducted an experiment with three users, u_A, u_B, and u_C, each having different devices and monitoring queries (see Figure 17). We investigate how cooperation planning is performed when the users come across, stay with, and leave each other. Figure 18(a) depicts four phases separated by the users' meeting and parting events. We show only u_A's viewpoint for concise description and verify the actual energy benefits. We set different cooperation policies for each user as: u_A

	u_A	u_B	u_C
Queries	Location, Dust, Ambient Sound	Ambient Sound, Dust	Location, Ambient Sound
Devices	Smartphone, Dust	Smartphone, Dust	Smartphone

Figure 17. Experimental setup



	PHASE 1	PHASE 2	PHASE 3	PHASE 4
Dust	Local (Dust sensor)	Exchange w/ Ambient	Exchange w/ Ambient	Local (Dust sensor)
Location	Local (Internet GPS)	Local (Internal GPS)	Co-monitoring	Co-monitoring
Ambient Sound	Local (Internal Mic)	Local (Internal Mic)	Local (Internal Mic)	Co-monitoring
Smartphone	596	613	316	365
Dust sensor	848	127	127	848

Figure 18. Experiment results for u_A

Table 3. Cooperation cases and expected benefits

Cooperation case		u_A's saving (mW)			u_B's saving(mW)	
ID	Description	Total	Phone	Dust	Phone	Dust
1	*case_co(ambient)*	30	30	0	12	0
2	*case_co(dust)*	336	-25	361	-24	361
3	*case_ex(u_A:dust, u_B:ambient)*	131	131	0	-5	127
4	*case_ex(u_A:ambient, u_B:dust)*	**704**	-17	721	**225**	0
5	*case_ex(u_A:ambient, u_B:location)*	231	231	0	-191	0
6	*case_ex(u_A:dust,* u_B:location)	454	454	0	-270	721

wants to maximize the total energy saving of the smartphone and the sensor devices, whereas u_B and u_C want to maximize the energy saving only for their smartphones.

Phase 1: u_A registers her location, ambient sound, and dust monitoring queries. As there is no cooperator, all those queries are processed by u_A's own resources. Figure 18(b) shows the power consumption of u_A's smartphone and dust sensor in Phase 1, i.e., 596 and 858mW, respectively.

Phase 2: Phase 2 begins when u_A meets u_B. Upon meeting each other, their CoMons start the cooperation planning process. By exchanging their demanding and suppliable contexts, both CoMons generate six cooperation cases and estimate the benefits for each case as in Table 3. u_A's CoMon determines that all cases are beneficial while u_B's determines that only *Case1* and *Case4* are so. Among the mutually beneficial cases, i.e., *Case1* and *Case4*, u_A selects *Case4*, *case_ex*(*ambient*, *dust*), as it is the most beneficial according to her policy. Now that u_A's monitoring queries are processed cooperatively, u_A's CoMon reduces the total power consumption from 1,444 to 740mW.

Phase 3: While u_A is being together with u_B, u_C comes across them. u_A's CoMon starts the cooperation planning with u_C as well, generating one cooperation case, i.e., *case_co*(*location*). Note that cooperation cases regarding the ambient sound context are not generated as it is already under the cooperation with u_B. u_A's CoMon begins additional cooperation with u_C by selecting and applying *case_co*(*location*); the total power consumption of u_A's devices has further reduced from 740 to 443mW.

Phase 4: u_B has just left u_A. Detecting the event, u_A's CoMon restores the dust monitoring to be processed by her own devices. Accordingly, u_A's total power consumption increases to 1,294mW. u_A begins additional planning with u_C on the dust and ambient sound contexts which u_A has cooperatively monitored with u_B. They generate and select a cooperation case, *case_co*(*ambient*), which is mutually beneficial. This new cooperation reduces the total power consumption of u_A's devices from 1,294 to 1,213mW.

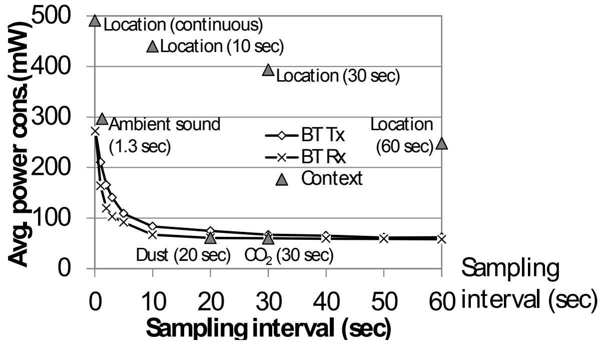


Figure 19. Exchange overheads and monitoring costs; the dust and the CO_2 sensor consumes 848 and 252mW, respectively

Cooperation between u_B and u_C: In phase 3, u_B and u_C also meet. However, any cooperation cases are not applicable because u_B had been already performing the cooperation with u_A for all the requested contexts, i.e., ambient sound and dust.

7.3 Cooperation Overhead

We examine the energy overhead for cooperative monitoring. We observe two major causes of overheads: (1) to discover nearby cooperator candidates, and (2) to exchange the monitoring results. We measure and analyze the overheads, showing that those are insignificant compared to the expected benefits.

Discovery overhead: CoMon conducts periodic Bluetooth scans for discovery, consuming additional energy. We measure the overheads for the scanning intervals in Section 4. The overhead is 20mW in our default interval of 5 minutes. This is relatively small compared to the expected benefits of many cooperation cases in Section 7.1. For example, if CoMon has been looking for cooperators for 60 minutes, it just needs 6 minutes to break even after starting the cooperation of *case_ex*(*ambient sound*, *location*).

Context Exchange Overhead: To identify the overhead for context exchange, we measure the smartphone's power consumptions for Bluetooth message exchanges as shown in Figure 19. To figure out the relative amount of the overhead, we also plot the smartphone's energy cost for monitoring several example contexts. For the contexts requiring power-hungry sensing or heavy computation, the energy overhead to exchange a context is much smaller than the cost to monitor the context. For instance, receiving the location context consumes only 60mW of the smartphone, whereas monitoring the context using GPS costs 333mW. In the case of *dust* and *CO_2* contexts, the smartphone does not benefit from the cooperation, but the sensor devices significantly save their energy as in Section 7.1. Note that some sensors such as accelerometers and illuminometer hardly consume power (<35mW) (less than the communication overhead for cooperation), and thus the cooperation might not be beneficial. These cases are filtered out through cooperation planning process.

7.4 Deployment

On top of the CoMon platform, we prototyped a proof-of-concept application, *TripMemory*. It is an Android application that tracks the user's travelling path and logs her surrounding events extracted from ambient sound [26]. We aim at presenting the effectiveness of cooperative context monitoring rather than showing the novel

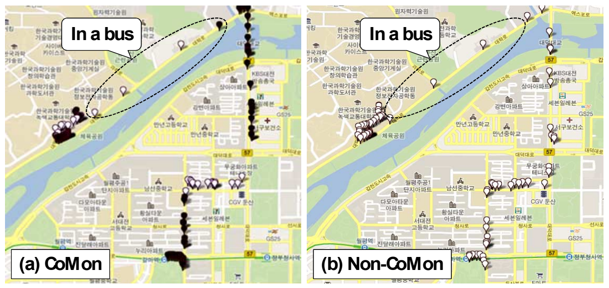


Figure 20. Location tracking from KAIST to the downtown

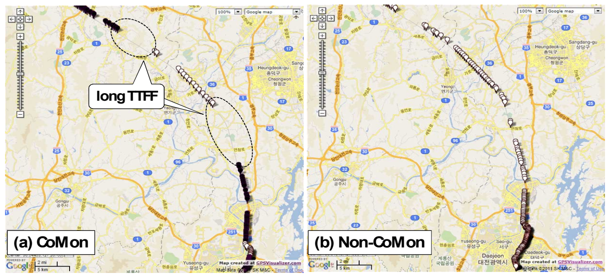


Figure 21. Location tracking in a high-speed rail

concept of the application. Upon the start of TripMemory, it registers either one or both of the following queries to CoMon requesting for user preferences; the query registration is performed once a day only.

CONTEXT location	**CONTEXT** ambient sound
PERIOD 5 Seconds	**PERIOD** 10 Seconds
DURATION Always	**DURATION** Always

CoMon notifies the application of monitoring results through the Android service interface. We recruited 12 participants consisting of 6 pairs of friends via the bulletin board of our school. Each participant was given a Nexus One phone with the CoMon platform and TripMemory installed. For comparison, each was given another phone with the same setting but deactivating the cooperation functionality of CoMon (named Non-CoMon). For fair comparison of the energy consumption, we used new batteries for the smartphones. The battery level of the smartphone is logged using Android library. We also required the participants to fully charge every night and not to run any application other than TripMemory. They roamed freely for a week.

According to our data, each participant runs TripMemory 6.2 days on average and 8.6 hours per day; some forgot to run for a day. On average, a pair cooperated 5.9 hours per day across 6.8 times of meetings. The average meeting duration is longer than we expected; we guess that this is because the participants are mostly close friends who are roommates or attending classes together. CoMon's average battery consumption is 19.7 % less than those of non-CoMon phones; this means that about 19.7% battery remains for a CoMon phone at the moment that the corresponding non-CoMon phone runs out of battery. Looking into the data, the cooperation benefits vary largely depending on the cooperation patterns. Only accounting for when a user turns on location monitoring, the benefits are 31.1 % in average. When both users turn on location and ambient sound monitoring, the benefits differ for the ones providing locations and the sound contexts. For the location providers, the average benefit is 6.9% only while the average benefit is 22% for the sound providers; note that location monitoring consumes a lot more energy. We expect that the benefit of CoMon will increase as CoMon is deployed by more people and more

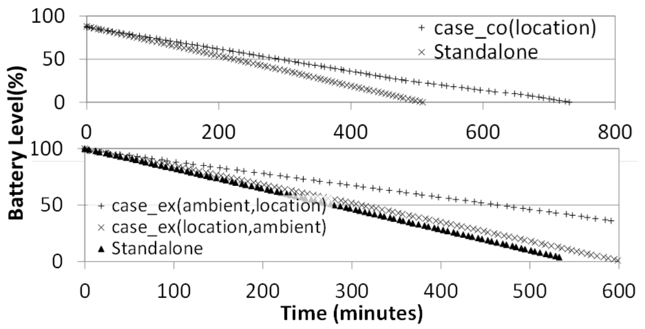


Figure 22. Battery level over time

Table 4. Stability results

Number of situations	Average duration (minutes)	Number of disconnection	Average disconnection time (minutes)
10	206	2.2	5.06

energy-intensive context processing is performed. In the rest of this section, we present interesting results and lessons from our deployment.

7.4.1 Quality of Application

To study the effect of the cooperation on the quality of the application, we compare the monitoring results from CoMon and Non-CoMon. We show a representative case in which a pair of participants, u_1 and u_2, trips together from KAIST to the downtown. During the trip, their CoMons performed *case_co*(*location*) and *case_co*(*ambient*) at the co-monitoring interval of 5 minutes. Figures 20(a) and (b) depict parts of the trip paths logged by TripMemory on u_1's CoMon and Non-CoMon, respectively; white dots represent the logs obtained from u_1's smartphone and black dots are those provided by u_2's. In the figures, the trip path recorded through the cooperation has very similar patterns to the path of Non-CoMon. It shows that CoMon does not compromise the quality of context monitoring much while achieving resource benefits. Note that different densities of the dots result from different transportation modes and constant monitoring period; they took a bus around the sparse part.

Figure 21 shows a rare but noteworthy case in which the monitoring quality was compromised. In the case, a pair of participants had an inter-city trip from Daejeon to Seoul by high-speed rail, running at a speed of up to 305km/h (190mph). Unlike the previous result shown in Figure 20, some location data are missing when switching the turns of co-monitoring. This is because it mostly takes a longer TTFF (time to first fix) as the device travels longer distance from the last detected location in its previous activation. We figured out that this problem happened when the user was moving extremely fast. It addresses two considerations that: (1) CoMon design needs to be further polished including a consideration of the delay to reactivate sensors, and (2) those delays of some sensors might be dependent on the some situational parameters, e.g., GPS sensor and the moving speed. We leave this issue for future work.

7.4.2 Long-term Observation of Battery

We present long-term observations of smartphone's battery level on TripMemory. We examine the battery behavior over time with the exemplary cases that the cooperation continues stably until the battery runs out. Figure 22(a) shows the battery level over time for co-monitoring of location information. CoMon extends the lifetime of the smartphone by 44% (from 508 to 732 minutes). Note that our benefit function and the offline profiling estimated the lifetime extension as 37% (see Section 7.1). We believe that the error comes from the fluctuations of the GPS and 3G signal strength, which affect the actual power consumption. Figure 22(b) describes the battery level when devices exchange the ambient sound context in return of the location and vice versa. *case_ex*(*location*, *ambient*) extends the lifetime from 532 to 598 minutes. *case_ex*(*ambient*, *location*) is more beneficial; the smartphone still shows 36% of the battery level at 598 minutes.

7.4.3 Cooperation Stability

We also investigate the stability of cooperation on top of CoMon using the TripMemory application. This shows that the stability of the connection is maintained while the users are cooperating, being physically close to each other. For this additional part of the experiments, we further recruited ten pairs of participants, where the participants in each pair are acquainted with each other. As a baseline, we have the users annotate the presence of their cooperators manually in case of disconnection and re-connection; we provide the users with vibrating and auditory alarms to notify the connection events. We collect the connection logs and the annotated data in diverse situations including dating, working in an office, and promenading along a river.

We look into the disconnection events and analyze them based on the participants' manual annotation. Table 4 shows the overall results. We first verified that there was no disconnection when they had been actually together. The intermittent disconnections occurred in our deployment situations were reconnected within 5 minutes on average, for example, the case of going to a restroom. This implies that, upon a disconnection from an acquaintance, waiting for 5 minutes might be worth trying, rather than immediately establishing a new cooperation with a stranger.

8. DISCUSSION

Coverage of Context Sharing. In the current design, we simply assume the range of Bluetooth (< 10m) as the coverage of context sharing. This works quite well in our deployment, where a pair of cooperators stays closeby during most of their meeting time. However, simply being within Bluetooth range does not ensure that two users have common contexts. For example, a user may detect another in the next room but may not have many common contexts. We believe this issue can be addressed in several ways. Exploiting Bluetooth RSSI [23] may deliver fine-grained clues on the inter-user proximity or the presence of obstacles separating them. Exchanging some contextual signature prior to cooperation may help to determine if they are in the same place. The place detection techniques such as SurroudSense [2] could be adopted for this purpose.

Privacy. Letting others know my context inherently raises privacy concerns. To be optimistic, we believe that such concerns might be relatively mitigated in the target environments of CoMon, where the users are physically in the same contexts. A study on location sharing supports that people are less conscious of sharing their locations when they are closeby [6]. A study on phone sharing shows that sharing is more acceptable with those in close social relations such as families, friends, or colleagues [24].

To be conservative, privacy concerns largely depend on different users and how the sensed contexts are to be used [19]. A user study implicates that people would be highly selective during their private time depending on their context and activities [5]. In this light, CoMon aims to provide users with the controllability and visibility on the sharing of their contexts. First, CoMon allows users to specify their sharing policies, i.e., the rules governing the access to their contexts from other cooperators as follows:

SHARE	Context
WITH	Target [EXCEPT Exception target]
RESOURCE	High | Middle | Low

CoMon provides simple UI showing the currently shared contexts and the cooperator information. We acknowledge that, the rules and UI address only some basic concerns on the privacy; it is still an open research question requiring further in-depth studies.

9. RELATED WORK

Collaborative Applications and Techniques. Opportunistic collaboration among smartphones has drawn attention in many domains, e.g., video playback and recording [3][33] and context inference [29]. CoMon takes collaboration opportunities for different purposes, e.g., saving energies for continuous context monitoring or obtaining new sensing modalities. Also, CoMon is the first to incorporate personal sensing devices into cooperation beyond smartphones.

Collaborative sensing techniques has been proposed to incorporate new sensing modalities and enhance data fidelity [10][20]. They share a high-level goal with CoMon in that it aims to increase the capability of individual mobile users through the collaboration. CoMon conducts its in-depth study on the cooperation opportunity and resource benefits of cooperation for continuous context monitoring.

There has been a research thread to develop middleware to construct mobile ad-hoc networks (MANETs) [4]. They group multiple mobile devices in an ad-hoc manner and enable applications such as file sharing and instant messaging. A major focus has been constructing network topologies through efficient discovery and advertisement of resources, such as mobile JXTA libraries. CoMon opens a new application domain of context monitoring, which clearly suits for the concept of in-situ sharing. In addition, different from conventional dynamic ad-hoc cooperation, we target continuity-lasting group of users to reduce the complexity and overheads for cooperation.

Participatory Sensing. This concept has been proposed to exploit the widely distributed mobile devices for urban-scale sensing applications. It has been adopted by many applications, e.g. pothole patrol [11], and has evolved into common platforms, e.g. PRISM [8] and Bubble Sensing [25]. These applications extend the spatio-temporal sensing coverage of a mobile user. Different from such works, CoMon aims to reduce the monitoring redundancies among the users in close proximity to resolve resource scarcity. CoMon is not a competing technology with participatory sensing but can complement each other. CoMon can serve as a client of participatory sensing, providing the contexts in greater energy efficiencies. In the other way, CoMon could utilize participatory sensing to extend its spatial context coverage.

Energy Optimization. There have been huge research efforts to reduce energy consumption for continuous sensing and data processing [16][30][34]. They focus on optimizing energy use within a single device whereas CoMon newly attempts to optimize resource use in consideration of multiple users and devices. A variety of techniques have been proposed, such as an energy-fidelity tradeoff [34], user interest-based sensor management [16], hierarchical sensor management [31], and low-power hardware mode utilization. Recent studies have delivered the energy efficiency for location monitoring [30]. CoMon complements such techniques for a single device, by further improving resource efficiency through active cooperation with nearby mobile users.

Task offloading as in MAUI [7] and Odessa [32] reduces resource consumption of smartphones; heavy back-end tasks in a processing pipeline are offloaded to servers. However, CoMon takes cooperation approach distributing tasks over nearby devices, having benefits not provided by the offloading approach. Many sensing tasks are not transferrable to remote servers since the sensing itself can be performed only where the context exists. Even for processing tasks, the overhead to transfer high-rate data could often overwhelm the resource benefit from offloading.

10. CONCLUSION

In this paper, we present the design and implementation of CoMon, a novel cooperative context monitoring system. We built CoMon by exploiting the prevailing cooperation opportunities among mobile users. CoMon allows every participant to take benefits from cooperation, through the continuity-aware cooperator selection and benefit-aware negotiation. We deployed CoMon prototype on off-the-shelf smartphones and diverse sensor devices. We evaluated that CoMon significantly improves resource efficiency for continuous mobile sensing and processing. Moreover, it extends the available contexts beyond those from one's own devices.

As people spend significant portion of time for social activities in their daily lives, smartphone applications and systems needs to be further evolved to support and leverage such situation. CoMon takes an initial step toward this direction, introducing social awareness into smartphones. It opens a broad spectrum of technical issues, e.g., networking for in-situ cooperation, co-activity group detection and managements, resource sharing and coordination, privacy. Based on CoMon and its underlying techniques, we are working on building a new platform to facilitate diverse in-situ social activities.

11. ACKNOWLEDGEMENTS

Our special thanks go to our shepherd, Professor Li-Shiuan Peh for her valuable suggestions to improve the quality of this paper. This research was supported by the NRF of Korea under grants No. 2011-0018120, SW computing R&D program of KEIT (2011, 10041313) funded by Ministry of Knowledge Economy of Korea, and partially by Microsoft Research Asia.

12. REFERENCES

[1] American Time Use Survey. http://www.bls.gov/tus

[2] Azizyan, M., Constandache, I., and Choudhury, R. R. SurroundSense: Mobile Phone Localization via Ambience Fingerprinting. In *MobiCom,* 2009.

[3] Bao, X. and Choudhury, R. R. MoVi: Mobile phone based video highlights via collaborative sensing. In *MobiSys,* 2010.

[4] Bisignano, M., Modica, G. D., and Tomarchio, O. JMobiPeer: a middleware for mobile peer-to-peer computing in MANETs. In *ICDCSW*, 2005.

[5] Choe, E. K., Consolvo, S., Jung, J., Harrison, B., and Kientz, J. A. Living in a Glass House: A Survey of Private Moments in the Home. In *UbiComp*, 2011.

[6] Consolvo, S., Smith, I. E., Matthews, T., LaMarch, A., Tabert, J., and Powledge, P. Location Disclosure to Social Relations: Why, When, & What People Want to Share. In *CHI,* 2005.

[7] Cuervo, E., Balasubramanian, A., Cho, D., Wolman, A., Saroiu, S., Chandra, R., and Bahl, P. MAUI: making smartphones last longer with code offload. In *MobiSys*, 2010.

[8] Das, T., Mohan, P., Padmanabhan, V. N., Ramjee, R., and Sharma, A. PRISM: Platform for Remote Sensing Using Smartphones. In *Mobisys,* 2010.

[9] Eable, N. and Pentlan, A. S. Reality Mining: Sensing Complex Social Systems. *Journal of Personal and Ubiquitous Computing*, Vol. 10, Issue 4, Mar. 2006.

[10] Eisenman, S. B. People-Centric Mobile Sensing Networks. Ph.D. Dissertation, 2008.

[11] Eriksson, J., Girod, L., and Hull, B. The Pothole Patrol: Using a Mobile Sensor Network for Road Surface Monitoring. In *MobiSys,* 2008.

[12] Google Android API. http://developer.android.com/guide/topics/wireless/bluetooth.html

[13] Gu, T., Pung, H. K., and Zhang, D. Q. A service-oriented middleware for building context-aware services. Journal of Network and Computer Applications, Vol. 28, Issue 1, 2004.

[14] iWake. http://itunes.apple.com/us/app/iwake-location-alarm/id406866935?mt=8

[15] Ju, Y., Min, C., Lee, Y., Yu, J., and Song, J. An Efficient Dataflow Execution Method for Mobile Context Monitoring Applications, in *PerCom*, 2012.

[16] Kang, S., Lee, J., Jang, H., Lee, H., Lee, Y., Park, S., Park, T., and Song, J. SeeMon: Scalable and Energy-efficient Context Monitoring Framework for Sensor-rich Mobile Environments. In *MobiSys*, 2008.

[17] Kang, S., Lee, Y., Min, C., Ju, Y., Park, T., Lee, J., Rhee, Y., and Song, J. Orchestrator: An Active Resource Orchestration Framework for Mobile Context Monitoring in Sensor-rich Mobile Environments. In *PerCom*, 2010.

[18] Kim, S. and Paulos, E. inAir: Measuring and Visualizing Indoor Air Quality. In Ubicomp, 2009.

[19] Klasnja, P., Consolvo, S., Choudhury, T., Beckwith, R., and Hightower, J. Exploring Privacy Concerns about Personal Sensing. In *Pervasive*, 2009.

[20] Lane, N., Lu, H., Eisenman, S. B., and Campbell, A. T. Cooperative Techniques Supporting Sensor-Based People-Centric Inferencing. In *Pervasive*, 2008.

[21] Lee, Y., Iyengar, S. S., Min, C., Ju, Y., Kang, S., Park, T., Lee, J., Rhee, Y., and Song, J. MobiCon: Mobile Context-Monitoring Platform. *Communications of the ACM*, 2012.

[22] Lester, J., Hartung, C., Pina, L., Libby, R., Borriello, G., and Duncan, G. Validated caloric expenditure estimation using a single body-worn sensor. In Ubicomp, 2009.

[23] Liu, S. and Striegel, A. Accurate extraction of face-to-face proximity using smartphones and Bluetooth. In *ICCCN,* 2011.

[24] Liu, Y., Rahmati, A., Huang, Y., Jang, H., Zhong, L., Zhang, Y., and Zhang, S. xShare: Supporting Impromptu Sharing of Mobile Phones. In *MobiSys*, 2009.

[25] Lu, H., Lane, N. D., Eisenman, S. B., and Campbell, A. T. Bubble-sensing: Binding SensingTasks to the Physical World. *Pervasive and Mobile Computing*, Vol. 6, Issue 1, Feb. 2010.

[26] Lu, H., Pan, W., Lane, N. D., Choudhury, T., and Campbell, A. T. SoundSense: Scalable Sound Sensing for People-Centric Applications on Mobile Phones. In *MobiSys*, 2009.

[27] Lu, H., Yang, J., Liu, Z., Lane, N. D., Choudhury, T., and Campbell, A. T. The Jigsaw continuous sensing engine for mobile phone applications. In *SenSys*, 2010.

[28] Milgram, S. The Familiar Stranger: An Aspect of Urban Anonymity. Addison-Wesley, Reading, 1977.

[29] Miluzzo, E., Cornelius, C. T., Ramaswamy, A., Choudhury, T., Liu, Z., and Campbell, A. T. Darwin phones: the evolution of sensing and inference on mobile phones. In *MobiSys,* 2010.

[30] Paek, J., Kim, J., and Govindan, R. Energy-Efficient Rate-Adaptive GPS-Based Positioning for Smartphones. In *MobiSys*, 2010.

[31] Park, T., Lee, J., Hwang, I., Yoo, C., Nachman, L., and Song, J. E-Gesture: a collaborative architecture for energy-efficient gesture recognition with hand-worn sensor and mobile devices. In *SenSys*, 2011.

[32] Ra, M-R., Sheth, A., Mummert, L., Pillai, P., Wetherall, D., and Govindan, R. Odessa: Enabling Interactive Perception Applications on Mobile Devices. In *MobiSys*, 2011.

[33] Shen, G., Li, Y., and Zhang, Y. MobiUS: Enable Together-Viewing Video Experience across Two Mobile Devices. In *MobiSys,* 2007.

[34] Sorber, J., Kostadinov, A., Garber, M., Brennan, M., Corner, M. D., and Berger, E. D. Eon: A Language and Runtime System for Perpetual Systems. In *SenSys*, 2007.

[35] Walkmeter. http://itunes.apple.com/us/app/walkmeter-gps-walking-stopwatch/id330594424?mt=8