Workshop Proceedings

International Workshop on SensorWebs, Databases and Mining in Networked Sensing Systems (SWDMNSS)

Workshop at the 4th International Conference on Networked Sensing Systems (INSS 2007)
Braunschweig, Germany
June 6, 2007

Yoh Shiraishi, University of Tokyo, Japan
Christian Decker, University of Karlsruhe, Germany
Koichi Yamada, Tokyo Denki University, Japan
Hiroki Saito, Tokyo Denki University, Japan

Workshop on SensorWebs, Databases and Mining in Networked Sensing Systems (SWDMNSS)

This workshop aims to discuss advanced technologies on SensorWebs, databases and data mining in order to tap new information resources of networked sensing systems and make them useable in a large variety of application contexts.

The development and deployment of sensor networking technologies brought the emergence of Internet-wide infrastructure for networked sensing systems and a collection of heterogeneous sensor networks. In this situation, a sensor network or a collection of distributed sensor networks is one of tremendous information resources on the Internet. This SensorWeb can provide real-time and historical information representing situations, contexts and changes in the real world. It is essential to tap this information resource in order to make networked sensing systems accessible and useable in a large variety of applications.

Sensor networks may be virtually regarded as an enormous database that can provide dynamic real-world information. There are great potentials in all areas of life, if people are able to query and search such information resources as they do it nowadays on the world-wide-web. In addition, the searched results should be provided in an understandable and potentially machine re-usable form. For instance, aggregated, summarized or symbolized sensor information is more useful for understanding complex phenomena than raw sensor information. Such information integration will be brought by novel methods for data mining and information extraction specifically tailored to networked sensing systems.

Also, information integration utilizing the world-wide-web may allow to combine sensor-specific querying techniques, e.g. according to geographic location, and web search engines, e.g. focusing on content, in order to approach an integrated real world search engine. In order to realize such real-world searching on networked sensing systems, some technologies from different research fields are required: a) network technologies to integrate and overlay heterogeneous sensor networks, b) database technologies to manage and search spatial and temporal information, c) mining technologies to extract useful information from sensor databases and other related resources.

This workshop aims to bring together technical papers about networks, databases and data mining for spatial and temporal information and discuss technical issues on integrating sensor networks, databases and Internet technologies. This workshop will open-up a novel and interdisciplinary research area for networked sensing systems covering different research fields such as networks and SensorWebs, databases, information integration and web engineering. It provides the opportunity to discuss key technologies for searching the real-world information provided by networked sensing systems.

Workshop organizing committee

- Yoh Shiraishi, the University of Tokyo, Japan
- Christian Decker, University of Karlsruhe, Germany
- Koichi Yamada, Tokyo Denki University, Japan
- Hiroki Saito, Tokyo Denki University, Japan

The organizer would like to thank the workshop program committee for supporting the review process with their expertise. The program committee includes the following experts:

- Michael Beigl, University of Braunschweig, Germany
- Christof Bornhoevd, SAP Research, Palo Alto
- Erik Hoel, ESRI, USA
- Takeshi Iwamoto, KDDI Labs, Japan
- Aman Kansal, Microsoft Research, USA
- Hideyuki Kawashima, University of Tsukuba, Japan
- Arei Kobayashi, KDDI Labs, Japan
- Antonio Krueger, University of Muenster, Germany
- Joseph Paradiso, Massachusetts Institute of Technology, USA
- Susanna Pirttikangas, University of Oulu, Finland
- Kazunori Takashio, Keio University, Japan
- Yosuke Tamura, Fixstars, Japan
- Yoshito Tobe, Tokyo Denki University, Japan
- Tomoki Yoshihisa, Kyoto University, Japan

The workshop organizers would also like to express their gratitude to the organizers of the INSS conference for providing the opportunity to hold this workshop. In particular, we like to thank the local arrangement chair Daniel Röhr for his excellent technical support.

Table of Contents Page

A Data Collection Scheme using Data Conjecture for Sensor Networks

Hozumi Kawanami[1], Tomoki Yoshihisa[2], Masanori Kanazawa[2], and Shojiro Nishio[3]

[1] Graduate School of Informatics, Kyoto University, Japan,
`kawanami@ais.sys.i.kyoto-u.ac.jp`
[2] Academic Center for Computing and Media Studies, Kyoto University, Japan,
`yoshihisa, bwv147@media.kyoto-u.ac.jp`
[3] Graduate School of Information Science and Technology, Osaka University, Japan,
`nishio@ist.osaka-u.ac.jp`

Abstract. To make weather-monitoring or see pictures of disasters, sensor networks, in which sensor nodes such as temperature or humidity sensors communicate with each other, have recently attracted great attention. In sensor networks, generally, a sensor node called the sink node collects sensor data generated by other sensor nodes. When there are too many sensor nodes, it takes long time to collect sensor data since the traffics for the sink node become large. Hence, in this paper, we propose a data collection scheme for sensor networks which have many sensor nodes. In our proposed scheme, the system conjectures several sensor data and broadcasts the conjectured data. By using broadcasting technique, the system can be durable for many sensor nodes. Our evaluations show that our proposed scheme can collect data faster than conventional schemes when there are many sensors.

1 Introduction

Recently, to make weather-monitoring or see pictures of disasters, sensor networks have recently attracted great attention[1]. Sensor networks ordinary consist of *sensor nodes* which are small computers with various sensors such as temperature or humidity sensors. Sensor nodes communicate with each other and sensor networks collect sensor data. For example, sensor networks are used for weather-monitoring[5] or visualization systems of temperature distribution.

Generally, sensor data observed by sensor nodes are collected by a specific sensor node called the *sink node*. Since each sensor node transmits its sensor data to the sink node, massive network traffics arise around the sink node. Therefore, various schemes to reduce the traffics have been proposed[2, 4, 6]. For the reason of page limit, we abbreviate the explanations of these schemes. These conventional schemes usually adopt server-client communication types. That is, clients transmit their sensor data to the sink node via the sensor network. For this reason, when there are a great number of sensor nodes, network traffics become high and it takes long time to collect sensor data. Cases that there are many sensor nodes follow:

- High dense sensing
  Accuracies of sensing are high as the distance between the sensor node and the target object is close. Therefore, a sensor network may allocate sensor nodes in a specific area to get high accuracy. In this case, there are many sensor nodes in a narrow area. In such high dense sensing, close sensor nodes may have similar value for the reason of sensor data locality.
- Wide area sensing
  The range that one sensor node can observe has a limit. Therefore, sensor nodes may be distributed widely to observe wide area. In this case, there are many sensor nodes over areas. In such wide area sensing, there are a little sensor nodes which have similar sensor data.

Due to the recent prevalence of sensor networks, it is supposed that sensor networks that have many sensor nodes will be practical. Accordingly, effective schemes to collect sensor data from many sensor nodes are required in sensor network research fields.

In this paper, we propose a data collection scheme for sensor networks which have many sensor nodes. In our proposed scheme, the system conjectures several sensor data and broadcasts the conjectured data to broadcast groups, i.e., groups of sensor nodes which have similar sensor data. Incoming sensor data are conjectured with a computer simulation. Generally, several sensor nodes have similar sensor data. For example, close sensor nodes in high dense sensing have similar sensor data. Therefore, the system can transmit the same conjectured sensor data to several sensor nodes by allowing slight errors. For example, suppose the case of distributing 100 temperature sensor nodes in a farm to find too cool fields for crops. They may be divided into 10 groups by each field. Since sensor nodes in a group are close, they have similar sensor data. Therefore, by allowing slight errors, the system can transmit 10 conjectured data to each group. The number of groups and the grouping algorithm is conducted by the administrators considering the system performance such as delay and power consumption. The amount of the admissible error depends on the application for the system. In the above example, we suppose that the system requires the accuracy of 1 degree Celsius for finding cool fields. Since sensor nodes know the real sensor data, the admissible error is assured by sensor nodes themselves.

Moreover, in our proposed scheme, by using broadcasting technique, the system can concurrently transmit the conjectured data to many clients[8]. Accordingly, the system can be durable for many sensor nodes. Sensor nodes that have sensor data over the admissible error from received conjectured data only transmit their sensor data to the sink node. For example, in the case where the difference between the observed real data and the conjectured data is more than 1 degree Celsius, the sensor node transmits the sensor data. Accordingly, network traffics are reduced and the necessary time for collecting sensor data becomes short.

The paper is organized as follows: We propose our scheme in Section 2, and evaluate it in Section 3. Finally, we will conclude the paper in Section 4.

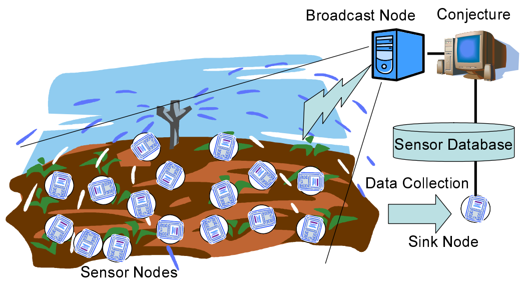


Fig. 1. Our assumed environment

2 Proposed Scheme

In this paper, to reduce the necessary time for collecting sensor data, we propose the Broadcast and Collect (B&C) scheme. In the B&C scheme, the system conjectures several sensor data and broadcasts the conjectured data. By using broadcasting technique, the system can be durable for many sensor nodes. In our proposed scheme, sensor nodes that have sensor data vastly differing from conjectured data only transmit their sensor data to the sink node. Accordingly, the network traffics are reduced and the necessary time for collecting sensor data becomes short. The name is derived from the fact that the data sampling interval consists of broadcast and collect phases in the B&C scheme as described later.

2.1 Assumed Environment

Our assumed environment is shown in Fig. 1. There are massive sensor nodes and they communicate with each other in their sensor network. The sink node collects sensor data from sensor nodes and it stores collected sensor data into the sensor database. These stored sensor data are used for conjecturing the incoming sensor data by the computer simulation. The broadcast node broadcasts the conjectured data to sensor nodes. The sink node and the broadcast node can communicate with other sensor nodes by single hop. The target time to collect sensor data is called *sampling point*, and the interval between sampling points is called *sampling interval*. Sampling interval is constant. Considering small computers with sensors, such as MICA[3] and mind storm[7], other assumed environments are show below.

- Sensor nodes cannot receive and transmit data simultaneously.
- Time Synchronization is completed and sensor nodes can recognize the current phase (broadcast phase or collect phase).
- Available bandwidth is fixed and stable.
- Sensor network itself has been constructed completely.

We show parameters for our proposed scheme in Table 1. The *broadcast group* is a group that includes several sensor nodes which have similar sensor data, i.e., they receive the same conjectured data. If the resource overheads are large, this corresponds to diminishing bandwidths, B_B or B_C. The hitting rate of

Table 1. Parameters for our proposed scheme

Symbols	Descriptions
N	The number of sensor nodes
M	The number of broadcast groups
D_B	The duration for the broadcast phase
D_C	The duration for the collect phase
B_B	The bandwidth for the broadcast node
B_C	The bandwidth for the sink node
H	The hitting rate of conjecture
S	The data size for one sensor data

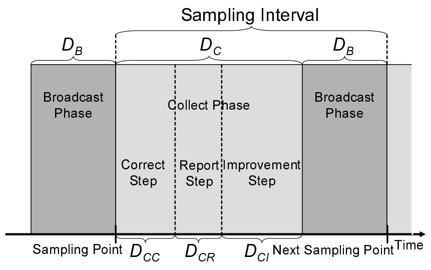


Fig. 2. Broadcast phase and collect phase

conjecture, H, is the probability that conjectured data are within the admissible error. When the accuracy of the computer simulation is low or the admissible error is small, H gets smaller.

2.2 Broadcast Phase and Collect Phase

In the B&C scheme, the data sampling interval are divided into two phases, the *broadcast phase* and the *collect phase*. This is because sensor nodes cannot receive and transmit data simultaneously. In the broadcasting phase, conjectured data are broadcast. In the collect phase, sensor data are collected. These durations are given by the system user. Fig. 2 shows these phases. The collect phase starts from the sampling point since sensor nodes have to compare the received conjectured data with the observed real sensor data. After the collect phase, the next broadcast phase starts. The broadcast phase and the collect phase are repeated every sampling interval.

Moreover, collect phase is divided into three steps. In the *correct step*, sensor nodes which observes the sensor data over the admissible error from the conjectured data transmit the real data to the sink node. In the *report step*, sensor nodes which does not receive the conjectured data transmit the real data to the sink node. Such nodes exist when the duration of the broadcast phase is short or broadcast error occurs. In the *improvement step*, the rest sensor nodes transmit the real data. Since the number of sensor nodes which transmit their sensor data in each step can be calculated probabilistically, the system can get the maximum duration of these steps. These three steps are summarized in Table 2. The correct step most efficiently refines the accuracy of data collection, and the report step

Table 2. Steps in collect phase

Conjectured data		Step to transmit data	Refining	Duration
Received	Over admissible error	Correct step	High	D_{CC}
	Under admissible error	Improvement step	Low	D_{CI}
Not received		Report step	Medium	D_{CR}

is more efficient than the improvement step. This is derived by the equation (1). Therefore, in the collect phase, steps are carried out in the order of the correct step, the report step, and the improvement step as shown in Fig. 2.

2.3 Adequacy and Reliability

In this subsection, we introduce adequacy and reliability. These values are used for evaluating our proposed scheme.

The adequacy is the probability of the sensor data that have values under the admissible error in the collected data. Since the probability of the sensor data that are under admissible error in the conjectured data is H, the value of the adequacy becomes more than H. Since the sensor data transmitted in the correct step and the report step refine the adequacy, the value of the adequacy is calculated by the following equation.

$$Adequacy = H + \frac{P_{CC}}{N} + \frac{P_{CR}(1-H)}{N} \tag{1}$$

The detail is abbreviated for the reason of the page limit. Here, P_{CC} and P_{CR} is the number of sensor nodes which transmit the observed real sensor data in the correct step and the report step.

The reliability is the probability of the observed real data in the collected data. The value of the reliability is equivalent to the rate of the sensor nodes transmitted sensor data in all sensor nodes. Therefore, the value of the reliability is calculated by the following equation.

$$Reliability = \frac{P_{CC} + P_{CR} + P_{CI}}{N} \tag{2}$$

P_{CI} is the number of sensor nodes which transmit the real sensor data in the improvement step.

Since the system user requires a larger adequacy and reliability, they can be indicators of the system performance.

3 Evaluation

In this section, we evaluate the performance of the B&C scheme. We compare it with conventional schemes which do not use the broadcast technique such as BBQ[4], Ken[2] and so on[6]. In such conventional schemes, the sink node collects sensor data for the duration of D_A. D_A is a parameter. Since $(1-H)P_A$

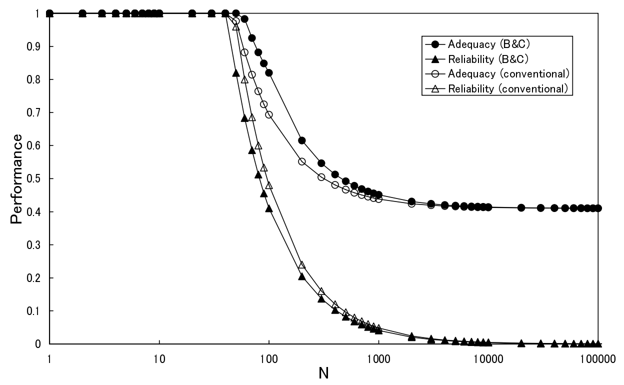


Fig. 3. Performances and the number of sensor nodes

sensor nodes have sensor data over the admissible error, the adequacy and the reliability for the conventional scheme are calculated by the following equations. Here, P_A is the number of sensor nodes which transmit the observed real data in the conventional scheme.

$$Adequacy = H + \frac{P_A(1-H)}{N} \tag{3}$$

$$Reliability = \frac{P_A}{N} \tag{4}$$

Results shown in this section are based on simulation results.

3.1 The Number of Sensor Nodes

We evaluated adequacies and reliabilities for the B&C scheme and the conventional scheme. In the evaluation, we set M to 10 and D_B to 0.017 sec. This is based on the example described in Section 1. Since the adequacy is maximum when D_B is 0.017 sec, we set so (for the reason of the page limit, we abbreviated the evaluation of D_B in the paper). Suitable parameters (D_B and D_C) are calculated easily by simulations. In this case, the sink node cannot broadcast the conjectured data to all broadcast groups and only 8 broadcast groups receive the conjectured data since the duration of the broadcast phase is short. In addition, we set D_C to 0.083 so that the sampling interval $D_B + D_C$ becomes 0.1 sec. Also, $D_A = D_B + D_C = 0.1$ sec. B_B and B_C are 250 kbps assuming that sensor nodes use the Zigbee protocol[9]. H is 0.5, and S is 64 bytes considering practical sensor data.

The results are shown in Fig. 3. The vertical axis is the adequacy and the reliability. The horizontal axis is the number of sensor nodes. "Adequacy(B&C)" is the value of the adequacy for our proposed method and "Reliability(B&C) is the reliability. "Adequacy(conventional)" and "Reliability(conventional)" are the adequacy and the reliability for the conventional scheme.

As shown in Fig. 3, the adequacy of the B&C scheme is better than that of the conventional scheme. This is because, since the broadcast node broadcasts

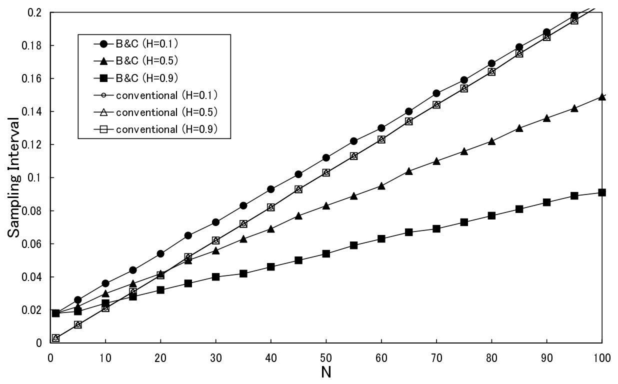


Fig. 4. Sampling rates and the number of sensor nodes

conjectured sensor data to sensor nodes, only sensor nodes which have sensor data over the admissible error from the conjectured data transmit their sensor data to the sink node. Therefore, the network traffics are reduced and the sink node can correct many sensor data. On the other hand, the reliability of the B&C scheme is worse than that of the conventional scheme. This is because the number of sensor nodes which transmit their observed data in the conventional scheme is larger than that of the B&C scheme since the conventional scheme does not include the broadcast phase. Also, the values of the adequacy and the reliability are 100% when the number of sensor nodes is less than 40. This is because the sink node can collect all sensor data since the number of sensor nodes is small.

For example, in the case where the number of sensor nodes is 100, the adequacy and the reliability of the B&C scheme are 90% and 40%. Those of the conventional scheme are 74% and 48%. Therefore, in the B&C scheme, we can say that 90% of the collected sensor data are accurate by allowing slight errors. The improvement ratio of the adequacy is $(90 - 74)/74 = 22\%$.

3.2 Sampling Interval

The system can require the 100% adequacy. For example, high accuracy weather-monitoring or the radioactivity monitoring. In Fig. 3, the 100% adequacy is achieved when the number of sensor nodes is less than 40. Therefore, we evaluate the minimum sampling interval that achieves the 100% adequacy. In the evaluation, D_B is 0.017 sec. This is the same as the previous evaluation. The results are shown in Fig. 4.

The vertical axis is the minimum sampling interval that achieves the 100% adequacy and the horizontal axis is the number of sensor nodes. "B&C($H = h$)" is the sampling interval of the B&C scheme when the hitting rate is h, and "conventional($H = h$)" is that of the conventional scheme.

As shown in Fig. 4, the sampling interval of the conventional scheme is shorter than that of the B&C scheme when the number of sensor nodes is small, and otherwise, the B&C scheme is shorter. This is because the broadcast technique effectively works when the number of sensor nodes is large.

For example, when there are 100 sensor nodes, the B&C scheme can collect data every 0.14 sec. In the conventional scheme, the interval is 0.21 sec.

4 Conclusion

In this paper, we proposed the B&C scheme to reduce the time needed to collect sensor data from the sensor network which has many sensor nodes. The B&C scheme conjectures sensor data and broadcasts them to several broadcast groups. Since the system can broadcast conjectured data to each broadcast group by allowing slight errors of sensor data, the network traffics are reduced and the time needed to the collection becomes short.

Our evaluations show that our proposed B&C scheme can collect data faster than conventional schemes when there are many sensor nodes. In the future, we will propose a scheme in the case where broadcast nodes are hierarchical or a scheme considering network delay by multi-hop communications.

Acknowledge

This research was supported in part by Special Coordination Funds for Promoting Science and Technology: "Yuragi Project" and in part by Grant-in-Aid for Young Scientists (B) numbered 18700085 from the Ministry of Education, Culture, Sports, Science and Technology of Japan.

References

1. Akyildiz, I. F., Su, W., Sankarasubramaniam, Y. and Cayirci, E.: A Survey on Sensor Networks, IEEE Communications Magazin (2002) 102–114
2. Chu, D., Deshpande, A., Hellerstein, J., and Hong, W.: Approximate data collection in sensor networks using probabilistic models, Proc. of the International Conference on Data Engineering (ICDE 2006) 48–59
3. Crossbow Technology, http://www.xbow.com/Home/HomePage.aspx (2006)
4. Deshpande, A., Guestrin, C., Madden, S., Hellerstein, J., and Hong, W.: Model-driven data acquisition in sensor networks, Proc. of the International Conference on Very Large Data Bases (VLDB 2004) 588–599
5. EXPO AMEDAS, an environmental data observation and display system, http://www.expo2005.or.jp/en/eco/amedas.html (2005)
6. Intanagonwiwat, C., Gvindan, R., Estrin, D., Heidemann, J., and Silva, F.: Directed diffusion for wireless sensor networking, IEEE/ACM Transactions on Networking, Vol. 11, Issue 1 (2003) 2–16
7. MINDSTORMS NXT Home, http://mindstorms.lego.com/ (2006)
8. Yoshihisa T., Tsukamoto M., and Nishio S.: A scheduling scheme for continuous media data broadcasting with a single channel, IEEE Transactions on Broadcasting, Vol. 52, Issue 1 (2006) 1–10
9. Zigbee Alliance, http://www.zigbee.org/en/ (2006)

An efficient data collection in sensor networks

Takahiro Ono[†,*], Ryohei Suzuki[††], Yoshito Tobe[†,*]

[†] Department of Information Systems and Multimedia Design, Tokyo Denki University
[††] Department of Information and Media Engineering, Tokyo Denki University
Email: {tak, ryohei}@u-netlab.jp, yoshito_tobe@osoite.jp
* CREST, Japan Science and Technology Agency

Abstract. Due to the advent of small wireless sensor network nodes, creating a large-scale wireless sensor network is becoming possible. If we apply conventional data-centric or in-network processing approaches to the networks, we will suffer from constructing per-node complicated rules. To cope with this problem, we introduce a declarative network approach to the sensor networks and thus simplifying the rules to be conducted at all nodes. In this paper, we show a simulation result based on the approach.

Key words: Sensor Networks, Data Collection, Declarative Networks

1. Introduction

Wireless sensor networks of the near future are envisioned to consist of hundreds to thousands of inexpensive wireless nodes which have some computational power and sensing capability. They are intended for a wide range of environmental sensing applications from vehicle tracking to habitat monitoring. The hardware technology for these networks such as low cost processors, miniature sensing and radio modules are developed currently, with further improvements in cost and capabilities expected within the next decade.

Wireless sensor networks are similar to traditional networks such as mobile ad hoc networks (MANETs)[1] in that both involve multi-hop communications. However, sensor networks differ from such the traditional networks in several ways: sensor networks have severe energy constraints, redundant low-rate data, and many-to-one flows. Additionally, increasing types of sensors may make the network be complicated. This is because that the increasing types of sensors cause in different sampling rate and various size of sensed data. Because of these aspects, traditional data collection methods are not suitable for sensor networks. To overcome these issues, we introduce declarative network[2][3] which utilizes database query processing technology into sensor networks for efficient data collection.

In this paper, we present a novel rule for efficient data collection in sensor networks. Researchers in declarative network focus on how to make routing table more simply and easily. However, they only give the policy of building the network. Because of this, they cannot utilize sensor network resource more efficiency. Therefore, we make an efficient data collection algorithm for sensor network in declarative language. It can be executed recursively at every node and collect sensing data each node. It looks like data collector collects the sensing data and takes into the request node which requires to obtain the sensing data. Actually, it happened with the declarative rule stored in every node in sensor networks. Based on these processes, we realize a simple architecture for large sensor networks.

The rest of this paper is structured as follows. Section 2, we present declarative network roughly. Section 3, we present our novel data collection design and execute that in sample network. Section 4, we evaluate our data collection algorithm based on the delay to measure the efficiency. Finally, Section 6 is conclusion of this article.

2. Overview of Declarative Network

Declarative network is a novel network architecture in which a distributed recursive query engine as a powerful vehicle for accelerating innovation. It is based on the foundation of database. This network is built with declarative language such as Prolog. It is used for network construction tool for building a declarative network. Since it is declarative, the result which is processing with this language is drawn deductively. This process is different from inductive one which uses some facts for making a result. Furthermore, compared with a procedural language such as java or C++, declarative language requires us less code. If you want to implement Chord[4] algorithm in C++, you need to write thousands of steps. In contrast, if you write that in declarative language, you only need to write 47 rules. So this language can be used to express a variety of well-known overlay network protocols in a compact and clean fashion, typically in a handful of lines of program code. Because of this point, we are released from writing a program for a long time. We can use more time for creating the new algorithm.

Actually, declarative language have many kinds of proposes in declarative network research, which depends on applications. In many cases, declarative language using for network is expanded model of Datalog, which is used in database query processing. If it applies to sensor network, it changes to SNlog[5]. On the other hand, it appears in Internet or Overlay network, it changes to NDlog. It is the most common declarative language for network. They are based on Datalog. Since the following, it looks like the same language, however, it includes location information means “at” sign (@).Based on these factors, nodes have own location in any other network, to specify the location is important.

Declarative network is accomplished the efficient utilization with simple and smart rules. This network supports our next generation network with the novel processing.

3. Design

3.1. Assumptions

We executed our data collection algorithm in declarative network which has different parameters and processing methods from traditional network. Although a data collection algorithm should be designed in a two-dimensional plane, we consider the basic structure in one dimension in this paper. We assume that the network is built with rules of declarative language based on the delay of each node. Every node has rule written in declarative language so that we make the network declaratively. Since data collection algorithm is the most important issue in sensor networks, we focus on how to collect sensing data more efficiency in sensor networks utilizing declarative methods.

As shown in Fig. 1, declarative network has differences from traditional network. For example, in the traditional network, if A requires data of B and C, A is required to send request packet twice. In contrast, in the declarative network, query is smart; the query allows the processing in the relay nodes. If A requires data of B and C, A is required to send query to collect data just one time.

Since the traditional network requires every node to collect data. Our data collection algorithm is shown in 3.2.

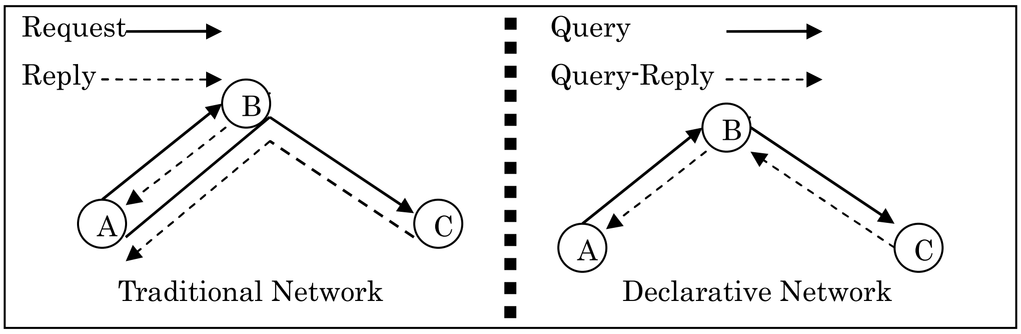


Fig. 1. Traditional vs. declarative networks

3.2. Data Collection Algorithm

Let us introduce our data collection algorithm in sensor networks utilizing declarative rule. Fig. 2 shows our data collection rules in declarative language. They have some arguments and functions to realize recursive query execution for an efficient data collection. Req, Rep, D, S, Table MyID and MyData is a request flag, reply flag, destination, a table includes node lists which is sorted by hop-count is utilized as a routing table in this network, a unique ID which is included in every node and a data which is a sensing data such as temperature or illumination, respectively. The ID which can identify the node, we call this node_ID. N is a next transfer node which is derived from function f_resolve_Next utilizing Table. Dv is a data vector which includes sensing data and node_ID to specify where this sensing data from is. Dv is the key variable to accomplish our efficient data collection algorithm. F_resolve_Next is utilized to specify the next transfer node which is the nearest node from the node in node list restricted same delectation to reply the request. f_make_Rep makes a Rep flag from Req flag. f_make_DataVector makes data vector (Dv) from ID and MyData. f_concat_Data fuses Dv, node_ID and MyData. ReqData() is the query from D which specifies a node which requires all sensing data. pathData() is the query to send the all sensing data to the request node. With utilizing these variables and functions, we make rules which are R1 and R2 in declarative style to archive an efficient data collection. These rules are included in every node to execute the declarative processing.

We show the execution with these rules in Fig.1. First, node A make a ReqData() query to collect data and transfer that to node B. If the destination of the query is not the B, the query transfers to the C. The C receives the query, and if the destination of the query is the C, the query applies to R1, after that creates N, Rep and Dv. These information are made to pathData() query and transfer this query to the B. If the destination of the query is the B, the query applies to R2, the Dv1 is fused with MyID and MyData to Dv with utilizing f_concat_data. These data and information made to pathData() query and transfer this query to the A. Finally, the A bring out data from the query.

Moreover, in a larger network as shown Fig. 3, R2 can be utilized more efficiency. R2 can collect the sensing data recursively, thus we do not need to write another rule for data collection. If we use a traditional algorithm, we need more than one rules for the communication pairs. Therefore, increasing the amount of data makes rule more complex than our process. In contrast, our rule can be utilized recursively; the number of node is not influences with complexity of data collection rules.

With this process, we accomplish efficient data collection in sensor network with the smart declarative algorithm. We evaluate this algorithm in our simulation environment.

```
R1: pathData(N, Rep, D, Dv) :- ReqData(MyID, Req, D),
                    N = F_resolve_Next(Table(N,,,,)),
                    Rep = f_make_Rep(Req),
                    Dv = f_make_DataVector(myID, myData)
R2: pathData(N, Rep, D, Dv) :- pathData(MyID, Rep, D, Dv1),
                    Dv = f_concat_Data(Dv1, MyID, MyData)
```

Fig. 2. Data Collection Rule

4. Evaluation

4.1. Emulation Environment

We make an emulation environment to evaluate our algorithm by simulation, we estimate a performance considering delay for data collection. After that, we compared the result using our suggestion with conventional algorithm.

We built network with ZigBee environment to evaluate our data collection algorithm and discover sensor network specific issues. Table. 1 shows bandwidth, delay and packet loss rate definition. The network is built with Free-BSD6.2 dummynet which is a network emulation system which can emulate restricting bandwidth, delay and packet loss. Fig. 3 shows the network which is utilized in our emulation. We conduct the experiments with this liner network by changing the number of nodes from 2 to 20. Network restrictions such as delay, packet loss and bandwidth concern the communication among with nodes every time. In this network, every node has dummy data such as DataA or DataB in Fig. 3 which is utilized as the sensing data in this emulation. In this environment, we measure the delay for data collection 10 times.

Table. 1. Emulation Environment

Operating System	Free-BSD 6.2
Compiler	GNU gcc 3.4.6
Communication	process communication
Network Emulator	Free-BSD dummynet
Bandwidth	250 [kbps]
Delay	10 [ms]
Packet loss rate	5 [%]

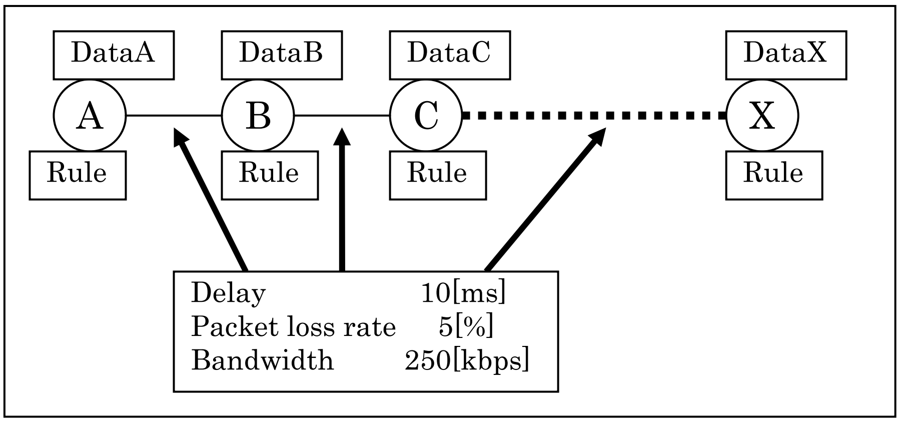


Fig. 3. Supposed Network Model

4.2 Emulation Result

Fig. 4 shows our emulation result with six graphs. In the figure, the average, the maximum, and the minimum of the measured values are shown. T and S represent traditional and our schemas, respectively.

As seen in the figure, our approach can reduce the delay for data collection. It is shown clearly when we increment the number of nodes. For instance, at 20 nodes in Fig. 4, the traditional technique needs 200 seconds to collect data by average. In contrast, our approach only needs 20 seconds. As a result, we can make sure that our data collection algorithm reduce the delay about 90 percents.

The reason of this result is that a traditional algorithm requires every node to collect data; a sink node needs to send queries to all nodes. Since a query is transmitted in a multi-hop way, increasing the number of nodes results in many relay nodes. In contrast, our algorithm can collect data recursively, thus we can collect data with one query to collect every data.

Since we conduct this emulation for a ZigBee environment, the bandwidth is set to 250 kbps. The fluctuation observed in Fig.4 is due to probabilistic behavior of packet loss. A Packet loss incurs retransmission bat the TCP larger and causes a larger delay. We assume that this TCP retransmission emulates a MAC-layer retransmission in a real environment.

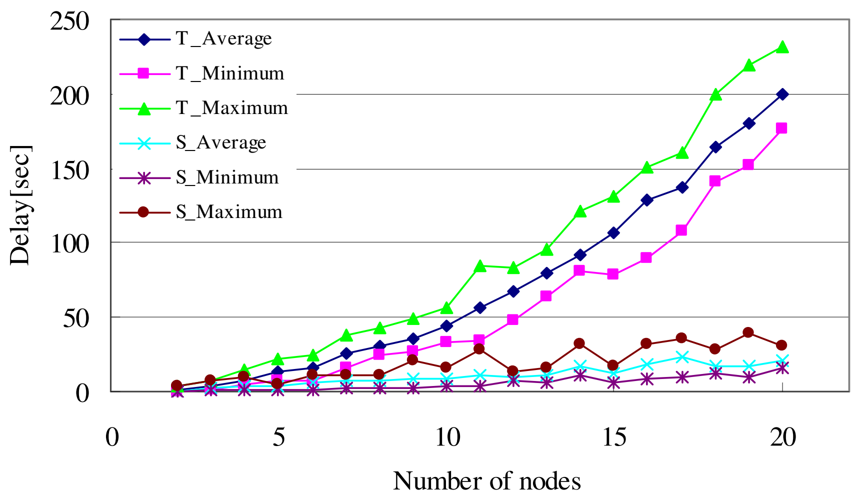


Fig. 4. Delay vs. number of nodes

5. Conclusion

We proposed an efficient data collecting scheme by utilizing a declarative network approach. An emulated result shows that our scheme effectively reduces delay for collecting all data and simplifies rules commonly installed onto all the words. As our future work, we plan to implement this scheme in our sensor network platform.

Reference

[1] M. Elhadef, A. Boukerche, H. Elkadiki, Diagnosing Mobile Ad-hoc Networks: Two Distributed Comparison-Based Self-Diagnosis Protocols. In MobiWac, 2006.
[2] B. T. Loo, Joseph M. Hellerstein, Ion Stoica, Declarative Routing: Extensible Routing with Declarative Queries. In *SIGCOMM,* 2005.
[3] B. T. Loo, T. Condie, M. Garofalakis, D. E. Gay, J. M. Hellerstein, P. Maniatis, R. Ramakrishnan, T. Roscoe, and I. Stoica. Declarative Networking: Language, and Optimization. In *ACM SIGMOD International Conference on Management of Data*, June 2006.
[4] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *SIGCOMM*, 2001.
[5] D. Chu, A. Tavakoli, L. Popa, J. Hellerstein, Entirely Declarative Sensor Network System. In VLDB 2006.

Mana: A Real World Oriented Query Processing System Supporting Control of Sensor Characteristics

Manabu Nakamura[1], Hideyuki Kawashima[2], Satoru Satake[1], and Michita Imai[3]

[1] Graduate School of Keio University
3-14-1 Hiyoshi, Kohoku-ku, Yokohama, Japan
{manabu, satake}@ayu.ics.keio.ac.jp
[2] University of Tsukuba
1-1-1 Tennodai, Tsukuba, Japan
kawasima@cs.tsukuba.ac.jp
[3] Information and Computer Science, Keio University
3-14-1 Hiyoshi, Kohoku-ku, Yokohama, Japan
michita@ics.keio.ac.jp

Abstract. For real world oriented applications, developers should consider data freshness, power consumption, expression of physical world and real-time response. However, controlling such sensor characteristics is a troublesome task. To overcome this problem, this paper presents a novel query processing system for real world oriented applications, Mana. Mana especially focuses on data freshness. The result of experiments showed that Mana satisfied data freshness compared with naive algorithm.

Key words: Data Freshness, Sensor Data, Middleware

1 Introduction

The advances of sensor devices such as RFID, wireless sensor nodes and cameras have made human living spaces recognizable. If intelligent robots or CG agents have ability to recognize the human living spaces, they would have ability to provide sophisticated services with interactions.

For these real-world oriented applications to use the result of recognition results, an infrastructure to manage conditions of physical objects obtained from sensor data is mandatory. Specifically considering, the requirements for an infrastructure are fourfold:

[R1] Providing the freshness of sensor data, because stale sensor data show the shadow of past physical world, which is not the current one and it makes real world oriented applications to misunderstand the current situation.
[R2] Reducing power consumption, because the battery of a sensor node is limited.

[R3] Expressing the physical world appropriately, because an excellent expression of the physical world helps users to retrieve data in the viewpoint of impedance mismatch.

[R4] Providing real-time response, because sensor data are generated periodically and a brand-new data should be transferred to a user before its freshness vanishes out.

Although all of these problems should be appropriately solved to realize an infrastructure for real world oriented applications, this paper focuses on problem **[R1]**.

For an infrastructures to support real world oriented applications, context model[3], MeT[1], semantic stream[4], and TinyDB[2] have been proposed. Context model describes and shares the environmental elements and the recognition relationship such as human/location/appliance. Therefore context model enables application designers to work without understanding the characteristics of sensor devices. MeT, a deductive database for sensor network, describes an environment by logical expressions. Thus the characteristics of sensor devices are abstracted and application designers can work without knowing the detail of sensor devices on ahead. Semantic stream divides recognition modules in the real world and enables to share them for applications. Each applications can ask to execute periodical monitoring to each module, and each module performs satisfying received requests. Therefore applications designers should not know the details of physical sensor devices. TinyDB considers a sensor network as a database system. It allows a user to set sensor data acquiring period, and therefore application developers should now consider how to acquire sensor data periodically.

Unfortunately none of the above researches considers problem **[R1]**. Context model does not consider data freshness. MeT does not consider data freshness at all. Semantic stream does not consider data freshness. TinyDB considers does not consider data freshness with starting point slippage. To the best of our knowledge, this is the first sensor database paper which focuses on data freshness issue considering power consumption, world expression, and real-time response.

This paper proposes Mana which is an infrastructure of a sensor network realizing **[R1]** on condition that **[R2]**, **[R3]** and **[R4]** are easily satisfied (e.g. targeting real world is not complex and data generation period is not frequent.). To satisfy **[R1]**, Mana conducts an algorithms:

As for **[R1]**, dynamically controlling the period of sensor data acquision for multiple continual queries and this algorithm selects the most efficient period of a sensor device saving power consumption in case multiple continual queries are issued.

The rest of this paper is organized as follows. Section 2 formulates problems. Section 3 proposes Mana. Section 4 evaluates Mana. Finally, Section 5 concludes this paper.

2 Preliminaries

2.1 Problem Formulation

The problem which this paper tackles is focused on [**R1**]. Stale sensor data show the shadow of past physical world, which is not the current one and it makes real world oriented applications to misunderstand the current situation of the physical world.

This paper defines the freshness of sensor data is the latency between its generation time and the arrival time to a client, and this paper denotes it just **latency** on below.

Each query requires a certain level of latency for each sensor data. Therefore, this paper formulates the first problem [**R1**] as [**P1**]

[**P1**]: Controlling the latency of sensor data under a requested value by queries.

2.2 Remaining Problems

The remaining problems are: [**R2**] reducing power consumption, [**R3**] expressing the physical world appropriately, and [**R4**] providing real-time response. They are also required by real world oriented applications, but we consider that in our targeting domain, they are easily satisfied as follows.

As for [**R2**], the intuitive and effective way to reduce battery consumption on a sensor node is aggregation of multiple queries. If n queries required to access a sensor s, without our optimization, n queries are executed on s in the worst case. While using our optimization, the number of queries which should be conducted is reduced to 1. Therefore the volume of battery consumption would be reduced to $1/n$ in the maximum case.

As for [**R3**], the physical world can be expressed as a tree model. The detail of this will be described in Section 3.1.

As for [**R4**], we assume that data generation period is larger than 1 second for each sensor.

3 Mana

To solve two problems formulated in the previous section, we present Mana. Section 3.1 describes the overview of Mana, and then Section 3.2 describes an approach for [**P1**].

3.1 Overview of Mana

Model and Query Language What real world applications require is the relationship between raw sensor data values and physical objects. For example, suppose a positional sensor p1 is attached to a cup c1. If a sensor data value obtained from p1 is (0, 10, 10), then this value should be associated with c1 such as "c1 is located at (0, 10, 10)". Without the association, sensor data are meaningless. From the rest of this paper we denote information associated to physical objects as object attribute. It includes position, temperature and luminous in this paper. Mana not only enables to search object attributes, but also specific information with an object such as name, height, or owner.

To realize an efficient search, Mana provides a tree structure model to express the physical world. In Mana, all of objects are classified in a tree structure as shown in Fig. 1. For example, to search all of tableware, cup, and spoon, a user is required to set tableware only. We understand there are more complex relationships in the physical world and it is an important problem as we referred it to [**R3**], but Mana limits the complexity to treat the problem easily, and it does not deal with such complex environment.

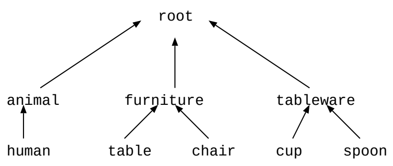


Fig. 1. Physical World Model in Mana

```
SELECT t.id, t.owner, t.pos_x
FROM tableware t
WHERE t.owner = 'manabu'
```

Fig. 2. Sample Query

For a query language, Mana provides a SQL like language. SQL is designed for relational database of which model is table. Since it is widely used and a variety of its extensions are presented, we adopted SQL for the basis of Mana's query language. An example is shown in Fig. 2. The example query searches ID, owner and x-axis position of tableware in condition that tableware's owner is "manabu". Plus, Mana allows a user to specify monitoring period in **RETRIEVE_RATE** clause, sensor data freshness in **FRESHNESS** clause, and query execution duration in **EXECUTE_TIME** clause.

Design and Implementation of Mana Mana is constituted of query processor, sensor period controller, synchronization controller, and environmental recognizer. The architecture of Mana is shown in Fig. 3. The result of recognitions is stored in a DBMS. Query processor is constituted of parser, query translator, and search executor. When a query is parsed, it is translated to a tree structure as shown in Fig. 4.

For the implementation of Mana, JDK 1.5.0 and almost 4000 lines are required. For data storage, we adopted PostgreSQL-7.4.6.

Deciding Sensor Period On receiving a query, Mana selects sensors to process the query as follows.

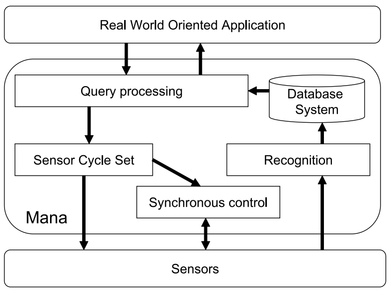


Fig. 3. Structure Of Mana

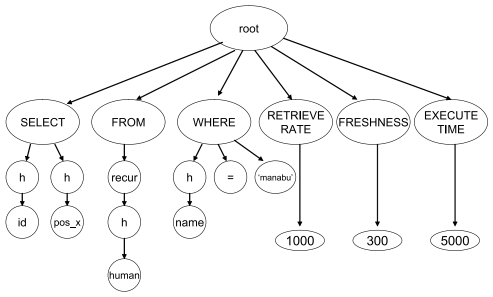


Fig. 4. Query Tree

1: Check whether a written attribute is derived by sensors or not. If it is inherent to an object, sensor are unnecessary.
2: Finding a set of sensor IDs to recognize specified attributes.
3: Collecting specified object IDs.
4: Selecting required sensors by the relationship between sensor IDs and object IDs.

For example, suppose pos_x of tableware is described in a query. In the first step, pos_x is checked whether it is derived by sensors or not, and then Mana detects it is derived. In the second step, a set of sensor IDs recognizing pos_x are collected and obtained as {s1, s2, s3}. In the third step, a set of IDs of tableware, cup and spoons are obtained as {t1, t2}. In the fourth step, necessary sensors are selected. If there are relationships that t1 recognizes s1 and t2 recognizes s2, then necessary sensors are obtained as s1 and s2.

After obtaining necessary sensors, Mana decides working period of sensors so that the period and the freshness of application requests are satisfied. Multiple requests can be generated by a query, and multiple queries can be issued by an applications. Therefore Mana should decides the period of sensors on this complex situation.

Mana considers power consumption of sensors, and it activates sensors in the longest period which satisfies application requests. The algorithm to decide sensor period is as follows.

1: Compare minimum period from all queries (min_p) with minimum freshness from queries except for the query with min_p (min_{ex_f}).
2: If $min_p < min_{ex\_f}$, min_p is set to the period of a sensor. Otherwise, go to the step 3.
3: Minimum requested freshness in all the queries (min_f) and the greatest common divisor of requested periods (gcd_p).
4: Compare min_f and gcd_p, and the larger one is set to the period of a sensor.

3.2 Synchronization Controller

Here we describe an algorithm to solve [**P1**]. Synchronization controller monitors sensors' working states and adjusts sensor data transmission time when latency is detected occurred by period starting point slippage.

To recovery the slippage, this module works as follows.

```
Algorithm 1: Synchronization Control (for [P1])

1: For each sensor, queries which require synchronization are obtained.
2: Queries which should synchronize freshness check are obtained.
3: For each query, freshness checks are conducted. If the freshness is not
   satisfied, then Mana issues a sleep request to a sensor.
```

In the first step, Mana obtains sensor working period and query freshness request. If sensor working period is larger, then Mana conducts synchronization control. The second step aggregates multiple requests to a sensor. Without this adjustment, same sleep requests can be issued separately nevertheless they are unnecessary. The third step issues sleep requests. This sleep request shifts starting point of a period to the ideal point.

4 Evaluation

The specifications of a machine for this evaluation are Linux Kernel-2.4.31 OS, Intel Pentium 4, 2 GHz CPU and 1 GB RAM. For the evaluation, we used five cups and each cup attached a positional sensor with an ID. The positional sensors are denoted as pos1, pos2, pos3, pos4, and pos5.

4.1 Synchronization Control for R1

Single Query Here we show how the synchronization control of Mana's important to satisfy data freshness. In this experiment, the following query is used.

```
SELECT c.id, c.pos_x FROM cup c
RETRIEVE_RATE 1000
FRESHNESS 500 EXECUTE_TIME 60000
```

This query retrieves the ID of a cup and its x-axis. To obtain the x-axis, positional sensor is necessary. Required period is 1000 ms, and required freshness is 500 ms. Since just only a query is executed, sensor works in a 1000 ms.

Since requested period is 1000 ms, for each 1000 ms interval, we measured latency with five sensors and checked whether required freshness (500ms) is satisfied or not.

Fig. 5 and Fig. 6 show the result of the experiments with synchronization control and without it, respectively. In these figures, x axis shows elapsed time, and y axis shows latency. The horizontal lines in the figures show the freshness

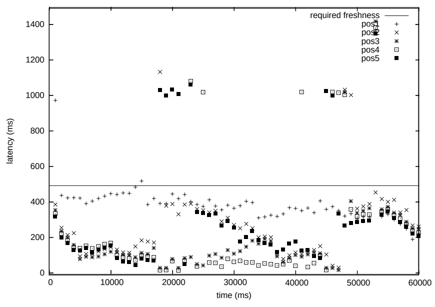


Fig. 5. Latency with synch. control

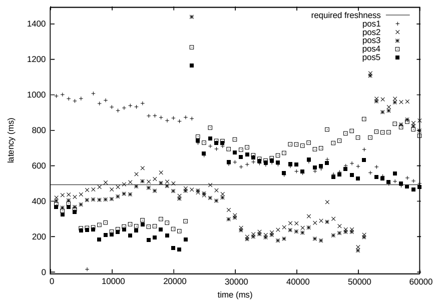


Fig. 6. Latency without synch. control

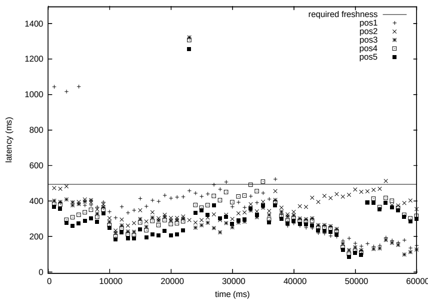


Fig. 7. Latency of Query 1

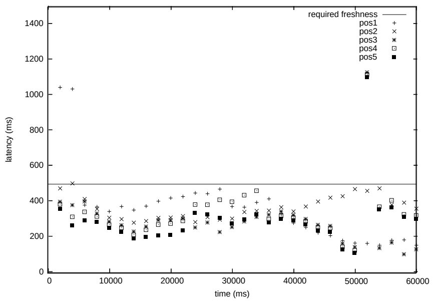


Fig. 8. Latency of Query 2

required by the query (500 ms). Dots over the line do not satisfy user request, while dots under the line satisfy. From those figures, obviously synchronous controller effectively works to satisfy user request.

In summary, Mana satisfied 92.3% with latency, while naive method satisfied only 46 %. Therefore synchronization control is consider to be effective for **P1** in case of single query.

Multiple Queries We also evaluated the synchronous control for multiple queries. We used the following queries. The difference between the two queries is retrieval_rate. The result of experiments with Query 1 and Query 2 are shown in Fig. 7 and Fig. 8 respectively. Obviously most of values are under the horizontal lines. Therefore Mana also satisfies user freshness level on multiple query environment.

In summary, synchronization control is consider to be effective for **P1** also in case of multiple queries.

```
Query 1:
SELECT c.id, c.pos_x FROM cup c
RETRIEVE_RATE 1000 FRESHNESS 500 EXECUTE_TIME 30000
Query 2:
SELECT c.id, c.pox_x, FROM cup c
RETRIEVE_RATE 2000 FRESHNESS 500 EXECUTE_TIME 30000
```

5 Conclusion

The problem which this paper focused on was "[**R1**]: controlling the latency of sensor data under a requested value". To efficiently solve it, a synchronization control algorithm was proposed. We implemented the algorithm onto Mana. The result of experiments and consideration showed that Mana effectively solved it. Furthermore, Mana satisfied the other three requirements [**R2**], [**R3**], and [**R4**] in a limited situation. Therefore, it should be noted that Mana can be applied for a kind of real-world oriented applications.

Although this paper seriously tackled only one problem, essentially, all the four requirements should be solved simultaneously. This is left as an open problem for the future work.

References

1. Hideyuki Kawashima, Yutaka Hirota, Satoru Satake, and Michita Imai. Met: A real world oriented metadata management system for semantic sensor networks. In *Proc. of the International Workshop on Data Management for Sensor Networks (DMSN*, pages 588–599, 2006.
2. Sam Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. Tinydb: An acqusitional query processing system for sensor networks. *ACM Transactions on Database Systems*, 30(1):122–173, 2005.
3. Ted McFadden, Karen Henricksen, Jadwiga Indulska, and Peter Mascaro. Applying a disciplined approach to the development of a context-aware communication application. In *Proceedings of the 3rd Int'l Conf. on Pervasive Computing and Communications (PerCom 2005)*, pages 300–306, 2005.
4. K. Whitehouse, F. Zhao, and J.Liu. Semantic stream: a framework for declarative queries and automatic data interpretation. Technical report, Technical Report MSR-TR-2005-45, Microsoft Research, April 2005.

An Assistance System for Safe Evacuation using Wireless Sensor Networks

Kenji Sasaki[†], Narito Kurata[‡], Nayuta Ishii [††], and Yoshito Tobe[†††]

[†] Department of Information and Media Engineering, Tokyo Denki University
[††] Graduate School of Advanced Science and Technology
[†††] Department of Information Systems and Multimedia Design,
Tokyo Denki University
‡Kobori Research Complex, Kajima Corporation
Email: kenji@u-netlab.jp, {nayuta, yoshito}@osoite.jp, kuratan@kajima.com
[Nayuta and Yoshito are co-affiliated with CREST, Japan Science and Technology Agency]

Abstract. Recently, large-scale earthquakes have occurred in many places in the world such as in Niigata, Japan and Sumatra, Indonesia. Importantly, second disasters may increase the number of damaged people. Providing correct information about evacuation can reduce the damage of earthquakes. Especially, information indicating the places with possible dangers is useful. We propose an assistance system for safe evacuation inside a building using wireless sensor networks called Risk Information Delivery System (RID). In this paper, we describe the design and the implementation of a prototype of RID.

Key words: Sensor networks, Monitoring of earthquake, Monitoring of fire, Inner-Processing

1. Introduction

We have recently seen much progress in the development of wireless communication and semiconductor technologies, which enables advanced technologies of sensor networks. Furthermore, new kinds of sensors have been developed by micro electro mechanical systems (MEMS) technology. In an architectural field, preventing various risks by using sensor networks beforehand becomes possible. There are structure automations[1], a structural monitoring[2], and safe evacuation in disaster area as utilizations of sensor networks. Especially, safe evacuation is expected as countermeasures of a large earthquake and an earthquake directly above its epicenter. In damages of the earthquake, there are the first disaster such as damages in a building and the secondary disaster such as fire in the structure. Damages due to a fire are the largest in the structure. Users might take evacuation by using a dangerous route because they cannot know where disasters occur after an earthquake. Therefore, an important problem is to offer users adequate information more immediately after earthquakes occur. In this paper, we propose Risk Information Delivery System using Wireless Sensor Networks (RID). RID detects dangerous areas by using a sensor network constructed in the structure and judges the degree of risk by calculating the risk of dangerous areas. The rest of the paper is organized as follows: Section 2 provides motivations of this work. Section 3 gives the design of RID. Section 4 describes the prototype of RID and the results experiment. Sections 5 and 6 present related work and conclusion and future works, respectively.

2. Background and Goals

The initial motivation for this work resulted from utilization of sensor networks for safety. Although smoke sensors and temperature sensors for detecting a disaster are currently available, they are not effectively cooperated. Therefore, an accurate observation of disaster situations is difficult. In this work, detailed sensing becomes possible by setting up nodes in a cooperative way. In addition, the cooperation of sensors becomes possible by converting all sensors into the same index of risk and users can obtain information recognized easily.

- Reliability of information

First, there is a problem that the number of packets increases when the flood of data is acquired from all the sensor nodes after earthquakes. Then, the raw data that each sensor node acquired is converted into risk information with a little volume of data. As a result, the number of packets decreases, and a dangerous part can be specified at early time. Moreover, a power consumption of each node can be decreased because of a decrease in the number of transmission packets.

- Reliability of wireless networks

Second, there is a possibility that the network becomes interrupted after an earthquake. Therefore, reconstruction of networks is necessary to while it does not influence the user's behavior as reliability of networks.

These two points are problems common not only to sensor networks but also to other wireless networks. However sensor networks have limitation of the battery drive and necessity of dealing with many nodes.

- Degree of risks

Finally, users need information that they understand in detail and momentarily. Therefore, we calculate degree of risk into which all sensor information is integrated by making the same index of risk. Moreover, this risk information is added to map information and offered as a map that the user recognizes easily.

3. System Design

This section describes assumptions for RID and its system architecture and functionalities.

3.1 Assumptions

We assume that all nodes are arranged everywhere in the structure. All nodes can belong to areas such as rooms and passages and communicate neighbor nodes. For sensors, we use sensors that can measure risk. These sensors include accelerometer, temperature, humidity, and smoke sensors. We assume that an accelerometer, and a temperature sensor. The accelerometer, detects earthquake and the temperature sensor detects fire.

3.2 *System architecture of RID*

The RID system is based on a client-server model and composed of RID_Client, RID_Server, Stationary node (SN) and Emergency node (EN) as shown in Figure 1.

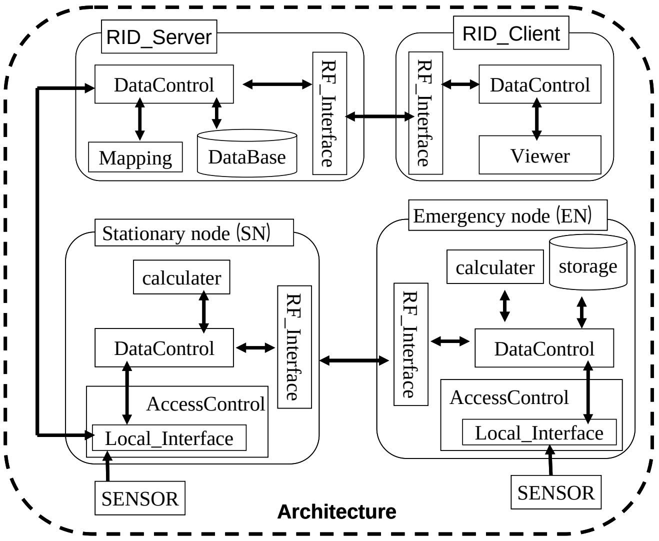


Figure 1 System Architecture

3.2.1 Stationary node (SN)

A Stationary node (SN) with an accelerometer always observes the state. When SNs detect the acceleration more than constancy, SNs judge occurrence of earthquakes and send wake up messages to Emergency Node. In addition, SNs aggregate risk information that ENs calculated and calculates risk of an area where SNs belong.

3.2.2 Emergency node (EN)

An Emergency node (EN) is equipped with sensors that detect various dangers. ENs start by receiving wake up messages from SNs by occurrence of earthquakes. Starting ENs send SNs risk information calculated from sensors data.

3.2.3 RID_Server

RID_Servers are equipped with SNs and save risk information on an area that SNs calculated in a database. Moreover, a risk map is constructed by adding risk information to map information. This map is offered to RID_Client.

3.2.4 RID_Client

RID_Client assumes PDAs and mobile phones carried by people. Viewer shows risk maps that RID_Server constructed.

3.3 Definition of areas and arrangement of nodes

We assume areas of narrow range such as rooms and passage as areas in structures. A SN and several ENs exist in these areas. SNs calculate areas of risk. Figure 2 shows the node arrangement example.

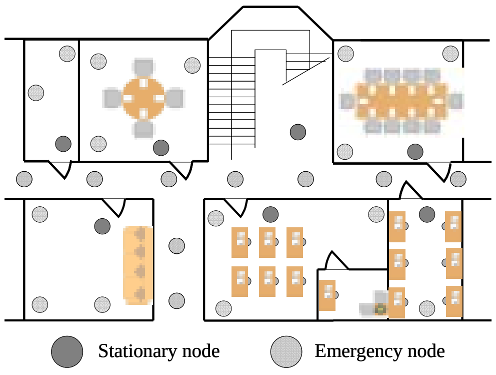


Figure 2 Example of arranging node

3.4 Restructuring of network according to situation

We consider the situation in which the node breaks down. A SN keeps transmitting connected packet to ENs at constant intervals while starting RID. ENs keep transmitting risk information to the SN while having received connected packet. Though it might cause connected packet to increase power consumption, the number of total packets that flows in the network including this factor will decrease obviously compared with the case to keep sending raw data. When the SN breaks down and connected packet becomes interrupted, ENs broadcast route request packets because ENs find a new SN. When a SN that belongs to another area receives route request packet, ID and the routing information that the node belongs are returned as route reply packet. Thus, a new route is constructed. When an EN breaks down, the degree of risk is calculated separating the node.

3.5 Calculation of risk

3.5.1 Definition of risk

We assume areas with the possibility that users receive harm to be dangerous and decide a size of degree of risk by making some thresholds in the value of sensors. The maximum value of degree of risk is 100, and a value changes based on the threshold. Table 1 and Table 2 show the degree of risk for temperature and the degree of risk for maximum acceleration. We referred to the standard of the constant temperature type warning machine that the Fire and Disaster Management Agency[5] provided for Table 1. The standard temperature in table 1 changes depending on the place and the climate. We referred to the material of the

Cabinet Office[6] for Table 2. A judgment of the degree of risk is defined by three stages. Users can take evacuation at the first phase from 0 to less than 20, need noting at the second stage from 20 to less than 60and must not act at the third stage of 60 or more.

Table1 Relation between temperature and degree of risk

Temperature	Degree of risk
Below a standard temperature	0
From 20 to less than 30℃ against a standard temperature	50
From 30 to less than 50℃against a standard temperature	75
50℃ or more from standard temperature	100

Table2 Relation among maximum seismic intensity, and degree of risk

Maximum acceleration [gal]	Seismic intensity	Degree of risk
0 ～ 40	～ 3	0
40 ～ 110	4	10
110 ～ 240	5 Lower	20
240 ～ 520	5 Upper	40
520 ～ 830	6 Lower	60
830 ～ 1500	6 Upper	80
1500 ～	7 ～	100

3.5.2 Calculation of the degree of risk in area

A SN calculates the degree of risk of areas. First, A SN aggregates risk information on all sensors of ENs. Next, the SN averages degree of risk information on each sensor and reflects the degree of risk of the sensor with the highest risk.

3.6 Construction of map

SNs are equipped with RID_Server. Because the risk map in each floor is offered, an upper and lower floor is not communicated. One of SNs becomes Sink when the system starts and risk information on each area is collected from SNs in the same floor through a wireless network. When Sink breaks down, other SN becomes new Sink and RID_server with the Sink constructs a map of a floor. For this case, Sink exists in each area divided into parts, and two or more maps are constructed in one floor. There is a possibility that networks are completely divided into parts awfully damaging the building though we are considered restructuring of networks as shown by 3.4.

4. Experiments and Evaluation

4.1 Assumptions of Evaluations

We implemented prototypes of both SN and EN and installed these devices on a shaking table. The shaking table that is shown in Figure 3 is 1.8m×1.5m and can shake at 0.1-100Hz using external input waves. A SN and an EN are Mica Z Motes. The sensor nodes utilize an accelerometer and a temperature sensor respectively to measure the degree of risk. Figure 4 shows Mica Z Motes and high accuracy accelerometers for comparison. We assume that the sampling frequency of these sensors is 100Hz. To evaluate our system performance, we did three kinds of experiment as follows.

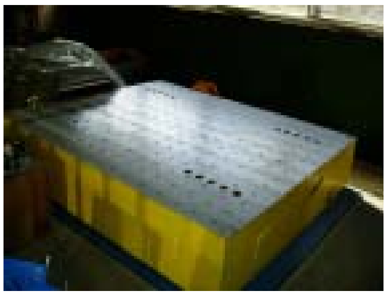


Figure 3 Shaking table

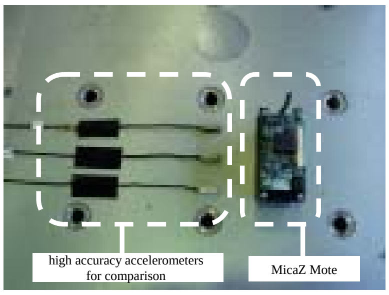


Figure 4 MicaZ Mote and high accuracy accelerometers for comparison

4.2 Experimentations

4.2.1 Performance of calculating the degree of risk

In this experiment, we install an EN on the shaking table. Then the EN is quaked on the table and heated by a drier. We collect all data (temperature, acceleration, risk degrees of temperature and acceleration) which an EN sensed during 40 seconds. We evaluate the ability of a measurement and the performance of calculating the degree of risk about the EN. Simultaneously, the maximal value of accelerations is averaged per ten samples and assigns the risk corresponding to Table1 using the average. On the other hand, sensed temperature data are averaged per one hundred samples every one second. Then our system subtract averaged temperature from the standard temperature that was measured when the system started and assigns the risk corresponding to Table2 using the averaged temperature. Figure5 shows measured data of high accuracy wired sensors for comparison. Figure6 and 7 show measured data of temperature and acceleration respectively. As the result of this experiment, our system can measure each degree of risk accuracy. Moreover the degree of risk is reflected on just timing against the actual measurement.

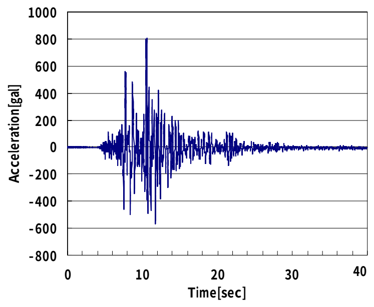


Figure 5 Result of a high accuracy accelerometers for comparison

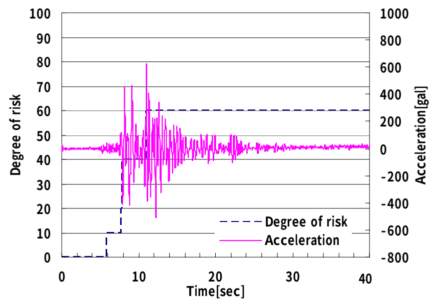


Figure 6 Result of an accelerometer with Micaz Mote

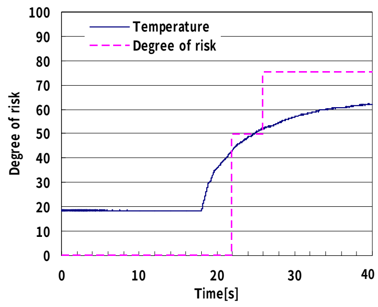


Figure 7 Result of a temperature sensor with Micaz Mote

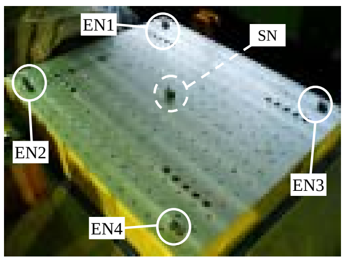
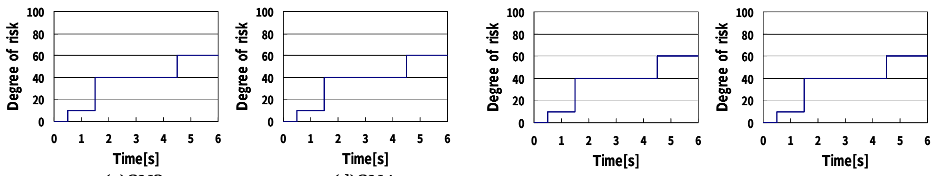


Figure 8 Appearance of arrangement of ENs and a SN

4.2.2 System performance

To evaluate our system performance, we consider shaking table to be one room and install a SN and four ENs on the shaking table. Then, we shake and heat sensors under the assumption that disaster occur. Figure8 shows mapping a SN and ENs on a shaking table.

First, a SN continues a sensing acceleration. If sensed acceleration exceeds the defined threshold, the SN starts to send a wake up message to ENs. We define the threshold as 20gal in this experiment. When ENs received the wake up message, these nodes start sensing acceleration and temperature and send only the degree of risk to SNs respectively every one second. Figure9 shows acceleration risk of all



(c)SN3 (d)SN4 (a)SN1 (b)SN2

Figure 9 Transition of acceleration risk of each EN

ENs from zero to six second that change was extreme. Accordingly, we found that four ENs sense same value in Figure9.

5 Related Works

There is a project of using a sensor network for measurement of the bridge. High performance three axis accelerometers were developed in this project. Moreover, the acceleration was measured by actually setting accelerometers up in the Golden Gate Bridge. The vibration always happens in the bridge because of the traffic of the car and the railway. Therefore, accelerometers can monitor a structural health. Wisden[4] aims at an efficient structural monitor, and collected data in a single server is supplemented with information on the storage of each node. Focuses of this research are a compression algorithm based on a wavelet transform and a same period of time.

Many researches that apply sensor networks to a structure focus on a measurement. RID has the feature in the point to grope, and to construct a wireless sensor network architecture with the application of risk offer. Additionally, RID converts into one information that unites all sensors, and constructs information that the user can momentarily judge.

6 Conclusion and Future Works

The purpose of this research is an application of sensor networks in the structure that considers the point of safe evacuation. We propose RID that detects a dangerous part in the structure and offers risk information after earthquake. People can get risk information on a detailed area such as rooms and the passages by RID. We mounted the prototype of a SN and ENs and experimented on performances. For our future work, we plan to experiment on the real world and to evaluate power consumption.

References

[1] R. Aipperspach, E. Cohen, and J. Canny, : Modeling Human Behavior from Simple Sensors in the Home, Pervasive 2006, (May. 2006)

[2] N. Kurata, B. F. Spencer Jr., and M. Ruiz-Sandoval,:Risk monitoring of buildings with wireless sensor networks, Structural Control and Health Monitoring, Vol.12, pp.315-327 (2005)

[3] Structural Health Monitoring of the Golden Gate Bridge
http://www.cs.berkeley.edu/~binetude/ggb/

[4] N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, D. Estrin, : A Wireless Sensor Network For Structural Monitoring, ACM SenSys 04, (November, 2004)

[5] The Fire and Disaster Management Agency
http://www.fdma.go.jp/

[6] The Cabinet Office
http://www.bousai.go.jp/

PATH: Phenomenon pAttern Template Handling - Design of an Application Framework for Sensor Networks -

Tatsuro Endo and Yosuke Tamura

Nisshin Building 3F, 1-8-27, Kounan, Minatoku, Tokyo, Japan
{endo,tamura}@fixstars.com
http://www.fixstars.com/

Abstract. Sensor networks are widely used in both civil and military applications, for example, security operation, surveillance, automation, and environmental monitoring. Typically, each sensor network is managed by a different party so, as a result, a variety of different approaches to monitoring have emerged. The purpose of this research is the integration of various types of sensor networks, and the generalisation of a scheme for accessing sensor information through the Internet. The main contribution of this paper is a dynamic drawing technique for establishing a correlation between a sensor and a physical phenomenon (an "event"). A web interface that depends on the sensors present could also be automatically generated.

Key words: Sensor Network, Web API, Application Framework, Web Application

1 Introduction

1.1 Background of Sensor Networks

Due to recent advances in Micro Electro Mechanical System (MEMS) technology combined with developments in wireless communications technology, Sensor Networks that can aquire various different types of event information have been receiving attention. Using MEMS technology, miniaturised sensor devices aquire sensor data and store it in databases using wireless transmission with little or no effect on the environment the data is taken from. As devices become smaller, and lower in power consumption, it is getting to the stage where they can operate for extended periods of time. The effect of this is that not only can we can also use them for long term aquisition of event information, but we can also use them for event cause analysis. It is now expected that sensor networks can be put to use in a broad variety of applications such as facilities management, examination of security information, and environmental monitoring.

However, as the development of Sensor Network applications has been dependent on the way event information has been needed to be aquired, arbitrary approaches have been taken to the way that the information in databases is aquired and searched. This means that, for applications incorporating sensor networks, every type of application has a different design. As previously mentioned, developments in sensor network technology are continuing and there are many areas in which these networks can be put to practical use, so research into the development of an application framework is necessary to meet the demand for efficient application development.

1.2 Internet services using Web API interfaces

We are proposing Web API as a method of accessing accumulated data from sensor networks. In recent years, Web API as been introduced as being geared towards users as a method of accessing the newest information on the Internet. Web API is an interface that enables the searching of data from every kind of enterprises. For example, Google currently offer a Web API interface for searching map information, Yahoo a Web API interface for searching intranets as opposed to the internet, and Amazon offer a Web API interface for searching current products. Through the introduction of these interfaces, website administrators are able to search every kind of enterprise from within their own websites. Through the use of Web API interfaces, applications could be developed that incorporate the ability to search all companies, and an increase in these new web-based services that can operate with multiple companies is expected. If sensor data could be could be accessed through a database of sensor networks using a Web API interface, one could similarly expect a new kind of web application that would allow access to this data more effectively over the internet.

However, as every company is developing their own form of Web API interface, a number of specifications are already in existence and at the moment they continue to flood in. The reason for this is not that every company has its own way of dealing with its products, nor is it that every company has its own data and database specifications; it is due to security concerns and the fact that every implementation is a black box. Multiple search capability is being hoped for, however when it comes to the intricacies of normal business dealings, one cannot expect the process of natural selection to occur just based on the merits of a Web API interface implementation alone. For the same reasons, standards convergence can also be seen to be difficult. For sensor networks that use Web API, similar problems will also arise.

1.3 Research Objectives

The aim of this research is to think about how information collected from sensor networks and stored in databases can be more effectively accessed through a sensor-network-oriented Web API interface. Using such an interface, many people could be able to use the same data in many different ways. Through solving the problem that is the flood of Web API interface specifications, we are aiming for a more efficient data access method. If it were possible to use Web API in a more integrated fashion, all of the aforementioned problems could be effectively solved. At that point this we would be expressing, through this paper, a more efficient method of using all of the numerous Web API interface specifications currently in existence.

2 Making Sensor Network data available through Web API interfaces

2.1 Dynamic Events in Sensor Networks

When obtaining event information from sensor networks, the relationship between an event and the corresponding sensor data can be both clear and not so clear. If the relationship is a clear one then it is not necessary for the application programmer to adjust the interface after it has been created. If the relationship is not such a clear one then one might have to make continual changes to the Web API interface. We need to handle this problem by generating the interface just once and then updating it automatically as an when required. Even in the case where the relationship is actually a clear one, one must assume that the interface will change as sensors will be added or removed from the sensor network from time to time.

If the framework were to automatically generate every Web API interface, though, we would then have a large number of Web API interfaces all with the same aim but none of which would be able to provide correct access to any data. A framework structure is necessary that allows people to make decisions as to whether interfaces are correct or not.

2.2 Generating Web API interfaces on top of Sensor Networks

From the standpoint of the development process, the burden of the development of applications for accessing sensor network data via the Internet could be reduced through the use of Web API interfaces. However, when developing applications for sensor networks, if one is hoping to aquire events where the number of sensors can vary then it is going to be difficult to assume that all applications will be the same when, for example, the specifications for the database housing the collected data, and the such like, are all different. The effect of this is that currently an application has to be developed for every event that one wants to investigate and every type of sensor that will collect those events.

Furthermore, if every user does develop a Web API interface, a similar large number of interfaces will exist in the same way that they do now and, amongst those, there will be some that will be trying access the same event data but using different specifications. In the initial stages, data access methods using the Web API methodology will reduce the burden for the application developer. However, it is forecast that, conversely, the burden will actually increase as handling the enornmous number of interfaces becomes extremely complex.

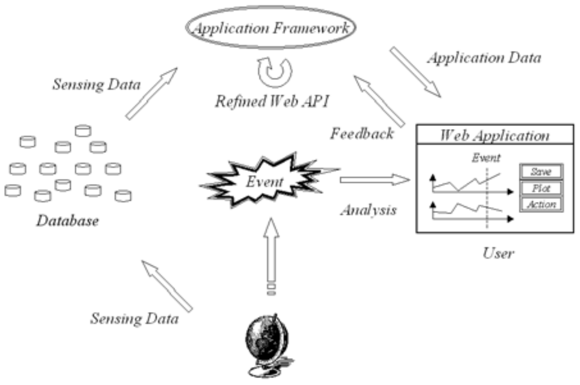


Fig. 1. System Overview including Application Framework

3 Design of an Application Framework for Sensor Networks

3.1 Unification of numerous Web API interfaces

Here we explain a method for unifying the huge volume of Web API interfaces that have come into being, a problem we mentioned in the previous section. If the relationship between the event a user wants to aquire and the sensor data is not a clear one then a situation will occur where the user creates a new interface that may not necessarily be correct. Also, there could be cases where a user creates their own interface that is the same as, but improves upon, another interface. From there it can be expected that a huge number of incorrectly defined, or duplicated, interfaces will arise. Additionally, where the number of sensors has been changed, it is possible that the interface may also change. It would be good if interface correction and unification could be done using both user feedback and repeated feedback from learning when trying to integrate and manage all the interfaces, even incorrectly defined interfaces, or multiple interfaces that have been created for similar purposes.

We consider interface learning as an application framework that, where a phenomenon arises that the user wishes to investigate, incorporates analysis of the sensor data, selecting the most appropriate interface based on the users consideration of sensor data, and updating the Web API interface. Within this framework, the correct relationship between an event and some sensor information is fed back via the user. Unification of interfaces would also be done and where as necessary where it is deemed that sensor data has been aquired in situations that are similar (and so data overlap has occurred). Figure 1 shows this concept. The globe represents all of the sensor networks that exist. The sensor data is saved in databases and is accessed by the application framework, which generates the Web API interface to publish the data. Users create applications that are based on the generated interface to analyse and display sensor data and

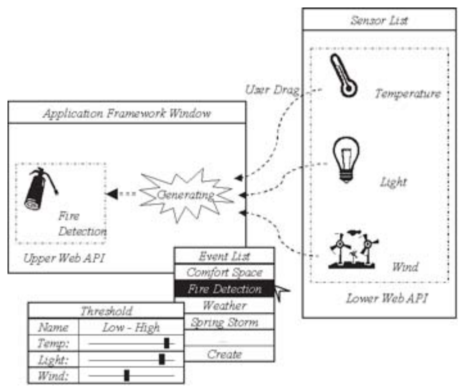


Fig. 2. User Interface for Application Developers

to infer events that have occurred. Through these applications, the user provides feedback on the interface, based on their knowledge of the event environment, which is then used to regenerate the interface for subsequent access of the sensor data.

3.2 An Efficient User Interface for the Web API Framework

In the preceding paragraphs we explained a method for unifying Web API interfaces where a large number of similar or incorrect interfaces have been generated. If you think about a sensor network acting as an infrastructure, interface unification through learning alone is sufficient, although even then it can still be expected that a very large number of interfaces will be generated. Below is the kind of method being thought about as a method to efficiently select a interface.

We consider an interface for reducing the burden of development for the application programmer by defining a framework in two layers. The lower layer consists of the original Web API interface for accessing the database, whilst the upper layer is an abstraction of the events that have been defined. The user selects an interface representing sensor data that is dependent on some event, and then tries, depending on the combination of sensors, to choose an event represented by an existing interface. In order to be able to choose an event we use a binary tree for all the separate cases. By defining constraints on the sensor data we can further narrow down the available events to choose from. In this case, a choice by the user from a list of applicable interfaces based on preprepared sensor data thresholds will identify the correct interface. Figure 2 shows an example of a user interface for this kind of framework. Application developers select sensor objects from a sensor list that is generated from knowledge of all available sensors in all known sensor networks. The framework generates the upper layer Web API interface (an abstraction of a particular event) from the available lower layer Web API interfaces which provide access to physical sensor data held in databases. In the UI, these lower-level interfaces could be represented as physical devices that

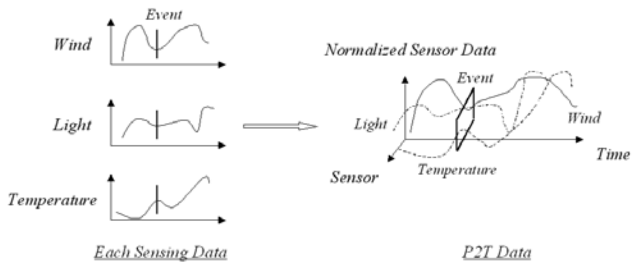


Fig. 3. An Example of Phenomenon Pattern Template

could be dragged into a "pool" area in order to combine their functionality. From this a list of events is generated that the user can select a single event from. By operating the threshold bars, constraints placed on the sensors, and the sensor data, will cause the list of selectable events to change and be regenerated or, in the case where the event has not been defined before, the user can cause a new interface to be generated to handle the specific combination of sensor types and thresholds that has been defined.

4 Phenomenon Pattern Recognition

4.1 Phenomenon Pattern Templates

In the previous section we talked about a Web API interface generation method that learns using Human Feedback. Here we will explain in detail how this learning is done. We mentioned about using feedback to find an event source from some event information, however it was expected that we would not be able to judge this just using the datestamps applied to the data at the moment each event was taken. For example, suppose we have a fire in the mountains that occurs as follows:

– Long spell of dry weather
– Leaves and other debris rub against each other due to the wind
– Leaves ignite due to friction
– Fire spreads due to the wind

If decisions are made just on discrete data points, it is difficult to determine the above pattern of events. Thus, in this research, we define the 3D Sensor Map. With a Sensor Type axis, a Normalised Sensor Data axis, and a time axis defined, as the sensor event time advances one can visualise the actual situation that occurred, and by plotting the data for each event one can build a 3D Sensor Map. With this kind of visualisation, making an objective decision becomes easier. Figure 3 shows an example of this mapping. The X axis is time, the Y axis is the normalised sensor data, and the Z axis going deeper into the page is the sensor type. Using this visualisation, discovery of the environmental changes with respect to time for the events becomes easier. Fig.3 shows the PATH of an example.

4.2 Phenomenon Pattern Recognition and Learning

With a sensor map one can obtain the relationship between a sensor datum and an event. One can judge which time intervals are considered to be important, and one can analyse the information before and after a change to an event. After that, using the normalised sensor data as a start point, you average the sensor data for an event at the time it changed with the data for the previous event. If you just take the average, a very large learning time becomes necessary and, where noise happens to be included in the sensor data, dealing with the averaging calculations becomes more difficult. So, we also assign a weight to each sensor datum with an aim to speeding-up the learning process by letting the user manipulate the weights to adjust the calculation of the averages. The learning equation is given by:

$$y_n = \frac{1}{n}\sum_{1}^{m} u_n + K(u_n - y_{n-1}) \tag{1}$$

$$K = \begin{pmatrix} k_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & k_m \end{pmatrix} \tag{2}$$

$$where$$

$$n > 1, m > 1, 0 < k_1, \cdots, k_m < 1$$

y is sensor data that has been learned, u is sensor data from database, K is a gain matrix, n is the number of sensor data samples, and m is the number of sensor data samples in a single time interval. All variables are defined as m dimension vectors.

We consider the time interval between events to always be the same. This calculation works for all sensors only and events within the overall selected time interval must be chronological.

5 Related Works

5.1 Pegboard

Pegboard[5] is a framework for mobile applications. It pays attention to visualisation and simplification in order to reduce the burden of development. To reduce the burden of physical code writing, it provides a good number of templates and sets out an Integrated Development Environment (IDE) where these templates can be chosen, and applications can be developed, just through various selections from within a GUI.

5.2 Semantic Streams

Semantic Streams[6] is a framework consisting of Inference Units and Event Streams and offers a declarative language for describing and composing inference rules for matching event stream requirements with sensors that can actually provide the event stream data. Using an algorithm called "Backward Chaining" to automatically select sensors from within a sensor network that are able to provide streams of a desired type of event, structures called Inference Graphs are used to store relationships between sensors and the inference units that process the data. These Inference Graphs enable similar resource queries to be resolved through query re-use thus optimising query processing and reducing overhead when resolving repeated requests for information from the same sensors. A real implementation of the Semantic Streams framework is described where a minimum number of sensors, installed in a car park, were able to simultaneously satisfy a group of different queries through re-use of sensor data, sensor processing code, and the sensors themselves.

6 Conclusions

In this paper, we describe an efficient method to acquire event information from sensor network on the Internet. We put together this paper as follows:

- The acquisition of event information from sensor networks.
- The definition of a dual-layer architecture Web API interface.
- A method of event pattern recognition, and learning.
- An efficient user interface for application developers.

In the future, we will develop an IDE that supports a Web API interface for sensor networks, and implements the interface learning method.

Acknowledgments. Japan Science and Technology Agency (JST) supported this research in Core Research for Evolutional Science and Technology (CREST) project. Members of Overlay-Network Search Oriented for Information about Town Events (OSOITE) that is a research in the projects helped us. Mr. Andre Sihera proofread this papar.

References

1. Google Code: http://code.google.com/
2. Yahoo Developer: http://developer.yahoo.com/
3. Amazon Web Services: http://aws.amazon.com/
4. World Wide Web Consortium: http://www.w3.org/2006/webapi/
5. Danny Soroker, Roman Caceres, Danny Dig, Andreas Schade, Susan Spraragen, Alpana Tiwari: Pegboard: A Framework for Developing Mobile Applications, The International Conference on Mobile Systems, Applications, and Services, June, 2006
6. Kamin Whitehouse, Feng Zhao, and Jie Liu, Semantic Streams: A Framework for Composable Semantic Interpretation of Sensor Data Proceedings of the 3rd European Workshop on Wireless Sensor Networks (EWSN 2006), February 13-15, 2006, pp. 5-20.

A Community Platform for Auto-Annotated Recreational Maps

Till Riedel[1], Phillip Scholl[1], Christian Decker[1],
and Michael Beigl[2]

[1] TecO,Universität Karlsruhe(TH), Vincenz Prießnitz Str.3
76131 Karlsruhe, Germany
<{riedel,scholl,cdecker}@teco.edu>

[2] IBR,Universität Braunschweig, Mühlenpfordstr.
Braunschweig , Germany
<beigl@ibr.cs.tu-bs.de>

Abstract. This paper proposes a novel way to generate and enhance recreational maps by using ubiquitous computing technologies. When planning trips one is often confronted with insufficient or costly map material. Maps are often not specialized enough for recreation or sport activity and personal fitness. Even if such maps exist they are out-dating very quickly as nature reclaims trails and damages road surfaces. As GPS devices and specialized sensor watches become popular tools for many mountain bikers, hikers and roller skaters, it makes sense to combine various information sources to automatically generate and share your ride on a track to create specialized recreational maps within a community of peers.

Keywords: ubicomp applications, sensor network, context awareness, location-based computing

1 Introduction

People commonly choose to be as far away from urban infrastructure as possible when performing their recreational activities. Maps and GPS navigation system are the nowadays tools to plan our outdoor activity. However, common freely available map material is in most cases not detailed enough or too outdated to be useful for detailed planning of mountain bike or hiking trips . Either it is focused on vehicle-access road infrastructure, which excludes the most interesting tracks for those sports or in the case of topological maps does not feature a important track properties.

Specialized hiking, biking or skating maps are needed, where the coloring and the style of the the route reflect the properties of a road on those map, which are strongly depending on the properties of the targeted sport.

Maintaining this specialized maps is costly while having small target audiences. This is due to nature's property of rather rapidly washing away former roads or simply roughening tracks just enough to be no more fun to travel by e.g. a bike. Therefore dynamic maps are needed that adapt to those changes dynamically.

In this paper we outline a method to automatically adapt map material by sensing the physical reaction of a sports device, in our example scenario a mountain bike, to a chosen track. Sharing sensor data gathered during multiple trips of different people in combination with their GPS track information enables the system to propose tracks to others with similar interest. We also describe a simple hardware prototype that can be used to gather the necessary data.

2 Related Work

Mapping sensor data to map material has become an actively researched area during the last years. The availability of sensor network data at many different locations have led to efforts like [3] to make shared data available through map interfaces. Those works mostly focus on the generic representation of data rather then support for specific applications and still have rather an academic focus.

In the area of sharing location data Internet communities like *http://www.gps-tour.info* already actively share their GPS track logs with others. More ambitious community projects like *http://www.maps4free.de/* even want to create free detail maps for GPS navigation. Those efforts show the high interest in freely available and documented map material.

From a research perspective [1] discusses in details the techniques and benefits of sharing GPS track log data to create recreational maps.

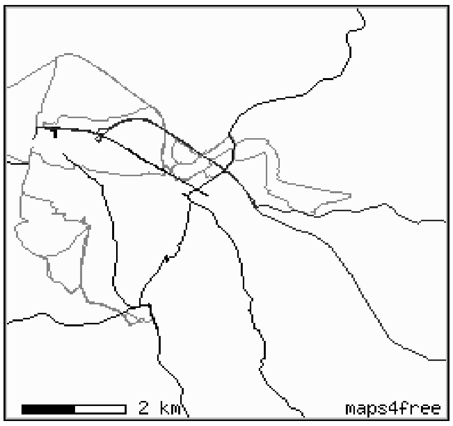


Figure 1: Web-based map server with track sharing

We believe that this is the right direction, however the usability of those services is limited as their route classification relies solely on manual annotation. Furthermore the annotation is either in such a loose format that it is difficult to extract data for personalized track planning or manual annotation and moderation becomes too difficult to gather the critical amount of data for reliable automatic map generation.

2 Automatically annotated maps

We claim that only an automated way of classification can generate a broad base of annotation data at decent quality. A critical mass of users can only be reached if the system can be designed in a way that involves reasonable effort for the user. This includes monetary cost for the necessary hardware, and what becomes more and more important: the cost of personal time. Especially in the area of recreation technology, the effort the user puts into the system should not exceed the immediate outcome. Otherwise the system contradicts the goal of recreation and will only reach a marginal audience.

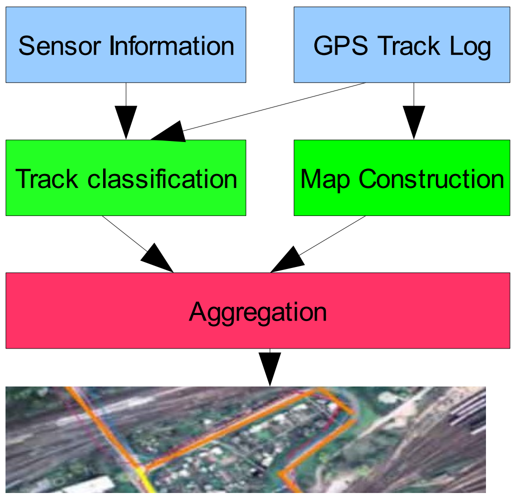


Figure 1: Overview of the proposed aggregation and clasiffication scheme

In this section we design a system that supports a community effort to generate personalized map material for its members in automated way.
A general overview of the design and the data flow of the system can seen in Figure 1. We identify three subproblems which will be addressed in the following sections:

1. how we classify the quality of the road by using sensor and location data.
2. how we build a map and decompose it classifiable elements
3. how we can use shared information to build a personalized map

2.1 Road Classification

Road classification can work in different ways. The simplest way uses data from other document-based data sources and reinterprets them in the context of the target system. A simple example is the route planner for bikes offered at http://www.viamichelin.fr that in some sense reverts the classification of automotive maps. Small roads are classified better to ride than highways. This, however, limits the search on roads accessible by cars. In contrast to this our approach uses a context recognition approach to allow track classification of previously unmapped roads.

We can take three types of context into account: the device context (type of bike, rollers skates, etc.), the personal context (the person gathering data and his physical fitness), and finally the road context (road quality) we want to extract.

The device context and personal preferences are partially known beforehand and can be configured by the user. In order to extract the remaining context sensors can be attached to the person's body and the sports device. This enables the system to automatically extract the quality of the road in respect to the person's fitness and his recreational goals.

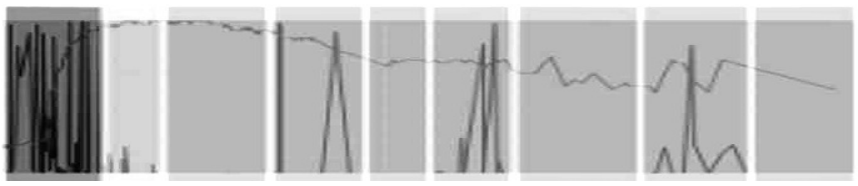


Figure 2: Classifying route quality by using vibration information

There are several possibilities of gaining such a classifications, which can be implemented by simple inference systems. As an example "if there are no shocks and low vibration the road is classified as smooth" and has therefore a "high quality for if one is riding a racing bike". If the rider is a mountain biker and likes a challenging experience those rules set might be replaced by its inverse.
Following this very simple reasoning in both cases the only property you have to measure is vibration. A simple sequence measuring vibrations and visualizing a possible classification is depicted in Figure 2.

In order to lower entry level complexity and to make the system attractive for new users, all rule set should can stored into profiles like "racing bike (intermediate level)", which can then be further personalized if the user feels the need to do so.

2.2 Map construction

By classifying track on the basis of a personalized preference we are able to annotate our own activities effectively. However, our work cannot end at the point of classifying a single track from a single person. Generating a valuable map mandates the availability of a hundreds of tracks in the same area best not yet traveled by the user before and still well annotated.

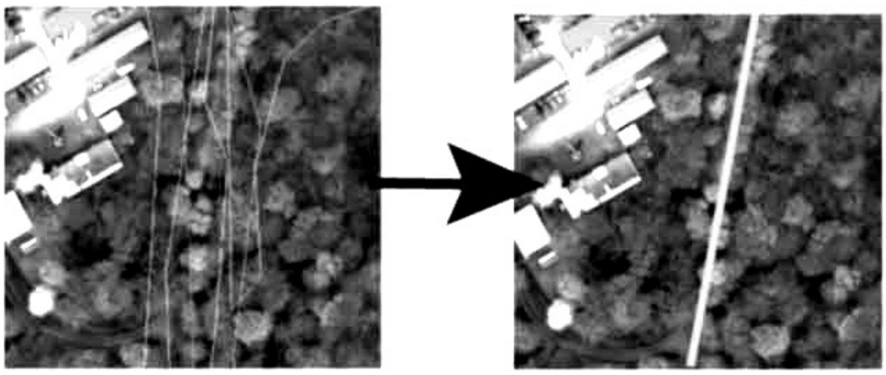


Figure 3: Unifying multiple tracks to a single map

This means that we have to use track information of different users and combine this into a map. The first problem, is that no two route are equal in a strict sense and therefore we would end up with a map that has tracks that differentiates the side of the road. That is why in the first step the proposed system needs to calculate the geo-spatial equality of tracks or parts of tracks, which is described in [1]. Figure 3 depicts how such an aggregation can look like.

Such a system works best if initialized with existing specialized map data. Therefore we can use manually annotated or unannotated data already gathered to initialize the system. By integrating more and more automatically annotated track data the often traveled routes soon will form a highly dynamical map. On top of this map a user can construct new routes for planning his next trip.

2.2 Personalization and Aggregation

Annotating the map on the basis of the per user track data is the next step in building a community-built map. The prerequisite is the separation of the data in classifiable elements and the clustering of groups of users with a similar understanding about road quality.

In our case we use track crossings to separate routes into classifiable part. If the goal is automatic personalized routing then it is sufficient to structure on this level. A routing algorithm only needs to make decisions at vertexes.

When inserting a new track we first match the track to the prerecorded routes on the map we reclassify the route parts on the basis of the new information. Therefore we take into account all parts of a route overlapping with the new track.

If we off-line classification methods instead of iteratively building the map based on aggregates, we have to save the parameters leading to the classification for all tracks and users for reclassification.

2.3 Sharing Community Platform

As mentioned in various places before the key element to a successful implementation of the system is a community portal that support the sharing of data and the access to maps in a convenient way. In our system this has to be enabled by a client for retrieving the data from both GPS receiver and sensor network. Existing standards like GPX can be used as a basis for data exchange.

The track qualification can easily be integrated as a scalar value into those exchange formats. By using a scalar quality measure we decouple the classification process from the aggregation and storage part. Furthermore we make sure that no privacy relevant data, such as heart rates is revealed. This also means that the user specific classification should be done on a personal computer instead on a central service. The preprocessed data can than be shared via the Internet.

Existing community platforms for GPS data have shown how successful community platforms can be built. The simplest model for such a platform is a central database with a web front-end for all track information. However, as spatial coordinates are globally unique and typical queries for recreational maps have a high locality it is easy to implement a scalable distributed version for the platform, e.g based on peer to peer technology.

3 Hardware Prototype

The second key element is hardware that can enable such a technology. For an implementation we suggest using sensor node hardware for sampling the vibration to measure vibration and other sport device specific features. The advantage of using small wireless sensing units is the possibility for easy embedding (see Figure x) without the necessity for special mounting and complicated wiring. Because of their small size and weight they can be attached to any part of the equipment without modifications. This ensures a high “portability” of the system to different equipment in terms of hardware.

The uPart platform [2] provides a prototype for such a ultra low-cost sensor network platform. The platform is based on low-power MCU which controls a send-only radio interface. It transmits data with an adjustable duty-cycle. It by default includes various sensors such as movement, temperature and light sensors but can easily customized to host other sensors. This design allows the node to run on a coin cell battery up to about 2 years depending on the duty-cycle.

The comparable low price of this platform also enables a higher coverage and redundancy of sensors on the sports device. Because normally the hardest part is the correct placement and calibration of sensor, it is better to place numerous sensors to detect the significant features for the classification.

In our initial design the data is received preprocessed and logged by a more powerful sensor node installed at a central location like the handle bars where attachment is less of an issue. This node has to mainly provide an amount of flash memory and a real time clock to time stamp the data. Such a platform can be the cPart that includes a CC1010 integrated MCU and radio transceiver that is inter-operable with the uPart platform.

The whole system also has to include a GPS receiver, which can but does not need to be coupled directly with the sensing hardware. Several specialized receiver types with things like map upload support are commercially available and are already available for many users. Using existing hardware therefore accelerate the adoption of our technology. Track data gathered from the GPS can be correlated with the sensing information via the timestamps on both systems instead of building an integrated system. This and all calculations are then done when connected to a PC and the Internet again.

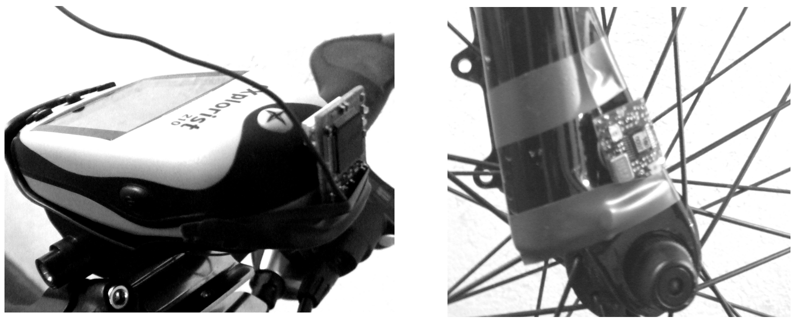


Figure 1: Shock sensor bundled with GPS on bike

4 Conclusion and Future Work

Overall we have outlined a system that can generate map material that is very detailed and up-to-date at very low cost. By collecting data on the spot we can directly visualize the effect of a road on a specialized activity. This way we generate a highly customized map beyond displaying mere cartographic features.

Because of the low entry-level investments both concerning personal and monetary effort, we believe that this technology can be adopted very soon by the growing group of GPS enthusiast.

6 References

1. Scott Morris, Alan Morris, Kobus Barnard, Digital Trail Libraries, *jcdl*, pp. 63-71, Digital Libraries, 2004 ACM/IEEE Joint Conference on (JCDL'04), 2004.
2. M. Beigl, A. Krohn, T. Riedel, T. Zimmer, C. Decker, M. Isomura, The uPart experience: Building a wireless sensor network, IEEE/ACM Conference on Information Processing in Sensor Networks (IPSN’06), ACM Press, ISBN 1-59593-334-4, pp 366-373, 2006
3. Suman Nath, Jie Liu, and Feng Zhao, "Challenges in Building a Portal for Sensors World-Wide," First Workshop on World-Sensor-Web: Mobile Device Centric Sensory Networks and Applications (WSW'2006)