International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE)
ISSN: 0976-1353 Volume 25 Issue 1 – JANUARY 2018.

Transmission of data through a MOBINET protocol to handle the duplication problem from the mobile sensor side

Punith Kumar[#1]

[#]*Guest Faculty, Department Of Computer Science, U.B.D.T College Of Engineering, Davanagere, India*

***Abstract*— Wireless Sensor Nodes (SNs), the key elements for building Internet of Things (IOT), have been deployed widely in order to get and transmit information over the internet. The sensor nodes are being deployed/installed on many objects and some of them are mobile (moving) including mobile gadgets, physical objects (living or non-living) etc. These mobile objects require sufficient Mobility Management Schemes to take care of data transmission. Host based mobility protocols; MIPv6 and its extensions are not suitable for these resource constrained devices. In this paper our focus is to study mobility management and different Scenarios based on it along with sensor devices. Existing research has made many improvements in terms of HO latency but less attention has paid towards signalling cost and packet loss particularly in time critical areas. The study provides the complete survey of network based mobility management schemes, challenges associated with them and solutions to meet these challenges.**

***Index Terms*—Mobility, Multihop, Mobinet, Level diagram**

I. INTRODUCTION

Observing the environment, tracking wildlife, tele-monitoring patients at home are some of the applications offered by a wireless sensor network. A wireless sensor has serious constraints such as limited and non-replenishable energy resources, limited memory and computational power. Networking them together raises new challenges in communication protocols. Recently mobility has been considered in wireless sensor networks.

II. PROBLEM STATEMENT

Today, there are plenty of routing protocols and it is very likely that two distinct wireless sensor networks use different routing protocols. With this hypothesis, it would be possible for mobile sensors to simply broadcast data to the fixed sensors. Upon reception, these fixed sensors will forward the messages to the sink. In this case, a mobile sensor does not need to implement the routing protocol used in the visited network. This solution, referenced to as broadcast method in the following, is a low-resource consuming solution for mobile sensors. Whenever a transmission is required a mobile sensor just needs to turn on its radio to transmit its message before going back to sleep. Consequently, the mobile sensor will never try to forward messages from the other sensors. This simple solution might seem ideal from the point of view of mobile sensors.

III. EXISTING SYSTEM

In multihop wireless networks, broadcast communications generally involve multiple receptions of each transmitted message. This may cause extra traffic, degradation of performance and may consequently increase energy consumption in the network. Instead of reducing such multiple receptions, the opportunistic routing protocols take advantage of them to increase the performances of the network. For example, the source attempts to deliver its message to the node (a relay), it is possible that this relay may not receive this message due to transmission errors (e.g. a collision has occurred). However neighbouring nodes may have received this message due to overhearing. Such message is generally discarded and the source continues to send its message until it is acknowledged by the relay. With opportunistic routing protocols, if neighbouring nodes do not hear such acknowledgement they try to organize themselves to forward a single copy of the message to the final destination. Therefore, the coordination between neighbors is the main challenge in opportunistic routing.

ExOR is the first proposition of an opportunistic routing. It uses a slotted version of the 802.11 MAC protocol and overhearing to schedule transmissions and prevent duplications. Before sending a message, the source selects a set of nodes located between itself and the destination according to the number of transmissions required to forward the message to the destination and includes it in the header of the message. This selection is enabled through a total knowledge of all edges of the network and their respective ETX values. Any node contained in this set is allowed to schedule a retransmission of the message. A node finds out when it is its turn to forward a message by overhearing other nodes which have a lower ETX (data forwarded or acknowledged). Scheduling by overhearing has few drawbacks, collisions on the medium or being unable to hear a message can affect the coordination of nodes and thus generate duplication. It is well known that centralized solutions have a prohibitive overhead, are hardly scalable and sensitive to topology dynamism. Thus, it is not a realistic

solution for wireless sensor networks.

CA-PATH is an opportunistic routing protocol developed for data gathering in wireless sensor networks. In order to reduce the energy consumed by sensor node, each node computes its ETX value toward the sink using local knowledge. This knowledge is gained by using the following scheme: first the sink periodically sends a beacon containing its own ETX value equal to 0. Any node overhearing this message updates its neighborhood table, computes the shortest path toward the sink and sends a message containing its own cost and the list of possible next hops. To schedule transmission between nodes, CA-PATH uses the same method as ExOR. The main drawback of this protocol is the complexity of the computation of the shortest path toward the sink. The authors decided to limit the complexity by using heuristics and only selecting the three next hops toward the sink. Since the coordination method is the same as ExOR, CA-PATH inherits the same drawbacks.

IV. PROPOSED SOLUTION: MOBINET

Mobinet consists in passively identifying the neighborhood of a mobile sensor in order to determine the best neighbor (according to the routing criteria of the protocol used in the visited network) to transmit data. The routing protocol of the visited network might use different criteria to forward data toward a sink: the next hop geographically closer to the sink, the next hop with fewer hops to the sink [9] or to the one whose battery has more energy.. Our solution fits intoa convergecast communication model in which the different devices are able to communicate at the MAC layer. The transmission of data through Mobinet is completed in two steps. The first step consists in building a neighborhood table (Section IV-A). Then the second step uses this table to select the next hop for the pending transmission (Section IV-B).

A. Network listening

When a mobile sensor enters into a visited network, it builds a neighborhood table that lists the identifiers of wireless sensors located in its vicinity. This table is constructed and kept updated by passively listening surrounding communications. The identifier of a sensor may be acquired, for example, using the source field of the MAC header of messages. Each entry of the table is associated with a time to live (TTL). When the TTL of an entry expires, this entry is removed from the table. This allows mobile nodes to remove old entries corresponding to neighbors that could be out-of-range.

The first trigger, named listen on TX, only wakes the radio up when a mobile sensor needs to send a message and waits for the interception of a surrounding communication. As soon as the mobile sensor has identified a neighbor and has successfully transmitted the message to this neighbor, the mobile sensor goes back to sleep. The second trigger, named table empty, ensures that the neighborhood table always have an entry and thus wakes the radio up when the last entry expires. The radio is kept up until a communication is intercepted. The last trigger is named periodic listening and activates the radio at regular interval to listen to the surrounding communications during a determined duration. If there is no entry in the neighborhood table while the mobile sensor wants to send a message, the listening process of Mobinet is switched to the first trigger. When the message is sent, the initial listening process is restored. The listening processes are detailed in Figure 1.

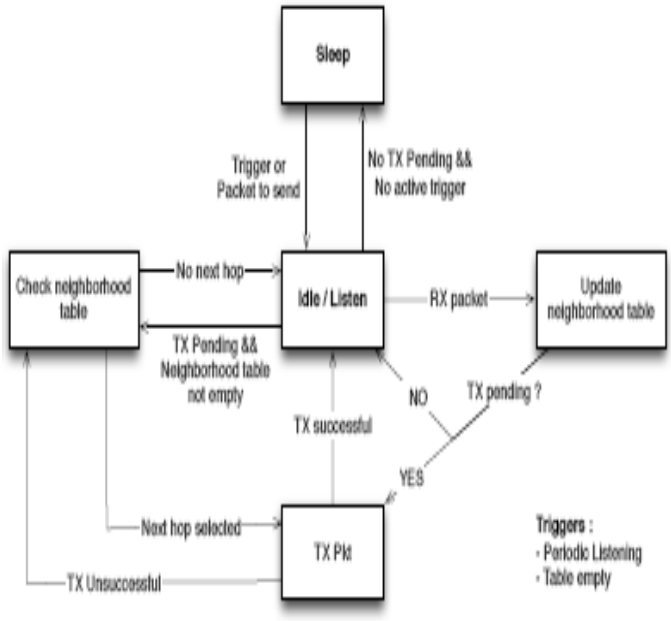

Fig.1. State machine diagram of Mobinet

With the great majority of MAC protocols, the mobile sensor needs to completely receive the message before being able to extract its source and its destination. Such overhearing may last for a long period of time and therefore may consume a lot of energy. However, this listening time can be reduced thanks to sampling MAC protocols such as X-MAC. XMAC uses several small preambles containing the necessary information to fill the neighborhood table. As a consequence, a mobile sensor just needs to listen to one of these preamble messages. Figure 2 illustrates the neighborhood listening at the MAC layer with X-MAC.

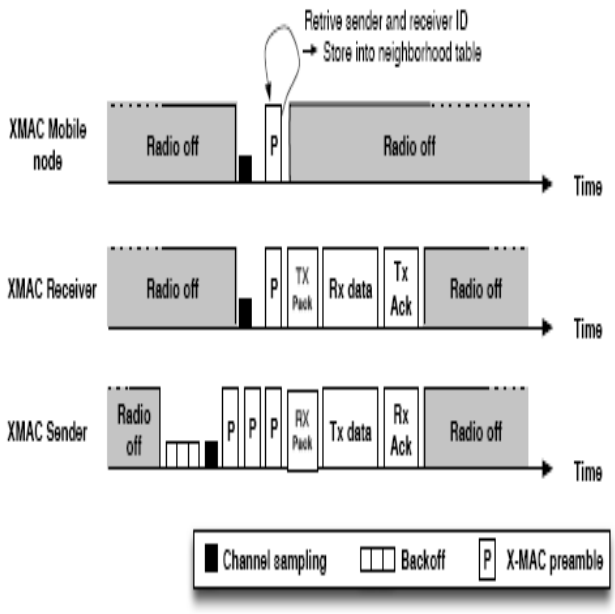

Fig.2. Neighborhood listening with sampling MAC protocols

B. *Next hop selection*

Once a mobile sensor wants to send a message to the sink of the visited network, it sends an unicast message to one of the neighbors listed in its neighborhood table. This is made possible by the convergecast communication model and the lack of routing header. This neighbor naturally forwards this message to the sink according to the routing protocol used in the visited network. In the first approach (referred to as random method hereafter), the neighbor selection is randomly selected among the ones available in the neighborhood. This approach is illustrated in Figure 3.

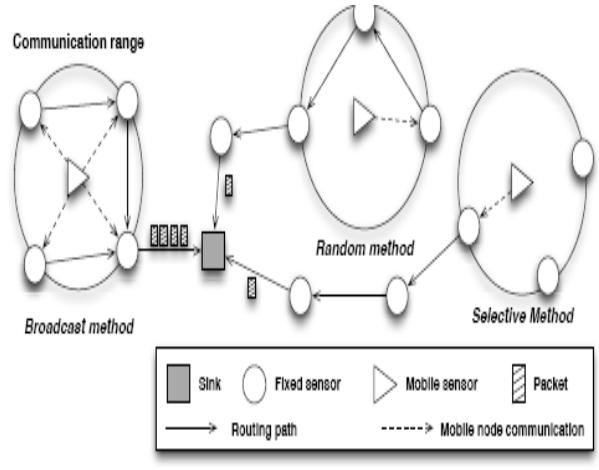

Fig.3. Next hop selection

From a longer passive listening, a mobile sensor can identify the direction of communications (from the source and destination of messages) so that it can create a hierarchy in its neighborhood table. Therefore, it should be able to transmit its data directly to the best sensor located in its neighborhood. The term best sensor is a generic term that can vary depending on the routing protocol used in the visited network (e.g. sensor that minimizes the number of hops to the sink). This second approach is referred to as selective method and is depicted in Figure 3.

Both approaches of our solution transmit data by establishing unicast communication. This avoids possible duplication of messages that is inherent to the broadcast method.

Simulation parameter	*Value*
Topology	Square grid (100 m x 100 m) of 100 (10x10) fixed sensors
Data sending period	Fixed sensor: upon an event, every second during 10 s Mobile sensor: every 4 minutes
Routing model	Gradient routing
MAC model	X-MAC with a preamble of 100 ms with acknowledgement
Radio model	Frequency: 868 Mhz ; Range: 15 m
Energy model with a 3V battery	Idle: 1.6 mA, Rx: 15 mA, Tx: 16.9 mA, Radio init: 8.2 mA
Number of events	3600 events
Duration and number of simulations	2 hours, simulated 100 times for each combination
Mobility	modified Random Direction model Max speed: 3 m/s
Mobinet	Neighbor entry TTL: 10 s

TABLE.1. SIMULATION PARAMETERS USED IN OUR EXPERIMENTS

Sim ID	*Listening process*	*Parameters*	
0	Broadcast method		
1	No listening process		
		Sleep	Listen
2	Periodic listening	1 s	20 ms
3	Periodic listening	1 s	100 ms
4	Periodic listening	10 s	20 ms
5	Periodic listening	10 s	100 ms
6	Periodic listening	60 s	20 ms
7	Periodic listening	60 s	100 ms
8	Table empty	Timeout: 10 s	
9	Listen on TX		

TABLE.2. SIMULATION ID AND LISTENING PROCESSES IN OUR EXPERIMENTS

V. REQUIREMENT SPECIFICATION

A. *Minimum Hardware Requirement specification:*

System : Pentium IV 1.8 GHz.
Hard Disk : 40 GB.
Ram : 512 Mb.

B. *Minimum Software Requirement Specification:*

Operating System : Windows XP.
Programming Language : C Sharp(.Net FrameWork)

VI. HIGH LEVEL DESIGN

A software product is a complex entity. Its development usually follows what is known as Software Development Life Cycle (SDLC). The second stage in the SDLC is the Design stage. The objective of the design stage is to produce the overall design of the software. The design stage involves two sub-stages namely:

- High-Level Design
- Detailed-Level Design

In the High-Level Design, the proposed functional and non-functional requirements of the software are studied. Overall solution architecture of the solution is developed which can handle those needs.

A. Design Consideration:

Predictive techniques are used to reduce the impact of dynamic topological changes within a zone. Location-awareness is key to determining the mobile nodes within a zone at a point in time. In our opinion location-aware routing is central to achieving proximity-bounded communication in a mobile network. We plan to extend this work to predict the future location of mobile nodes. Using this information future node movement into a zone, the impact on routing, resource reservation, and guarantees for timeliness and reliability for the zone can be predicted in advance.

The ability to predict node movement contributes to achieving probabilistic guarantees of path availability due to link failure caused by node mobility. Other reasons why a link may fail, such as environment conditions or battery usage, must also be considered to avoid or anticipate network partitions. Using partition anticipation based on coupled with proactive and preemptive routing and resource reservation, we aim to improve the rerouting process by attempting to find new paths prior to the failure of existing ones. To obtain mobility independent real-time guarantees, a mobile host would need to make advance resource reservations at predicted locations they may visit during the lifetime of the communication. Accurate mobility and location prediction is critical for limiting the overhead of excessive resource reservation.

B. Development Methods:

The development method used in this software design is the modular/functional development method. In this, the system is broken down into different modules, with a certain amount of dependency among them. The input-output data that flows from one-module to another will show the dependency. Data flow diagrams have been used in the modular design of the system. Structure charts have been used to show the hierarchy of the function calls in the system. Data Flow Diagram and structure charts are mentioned.

C. 6.2 Data Flow Diagrams

Data-flow models are an intuitive way showing how data is processed by a system. At the analysis level, they should be used to model the way in which data is processed in the existing system. The notation used in these models represents functional processing, data stores and data movements between functions. Dataflow models are used to show how data flows through a sequence of processing steps. The data is transformed at each step before moving on to the next stage. These processing steps or transformation are program functions where dataflow diagrams are used to document a software design.

With a dataflow diagram, users are able to visualize how the system will operate, what the system will accomplish and how the system will be implemented. Old system dataflow diagrams can be drawn up and compared with the new systems dataflow diagrams to draw comparisons to implement a more efficient system. Dataflow diagrams can be used to provide the end user with a physical idea of where the data they input, ultimately has an effect upon the structure of the whole system from order to dispatch to restock how any system is developed can be determined through a dataflow diagram. There are several common modeling rules to be followed while creating DFDs are as follows:

- All processes must have at least one data flow in and one data flow out.
- All processes should modify the incoming data, producing new forms of outgoing data.
- Each data store must be involved with at least one data flow.
- Each external entity must be involved with at least one data flow.
- A data flow must be attached to at least one process.

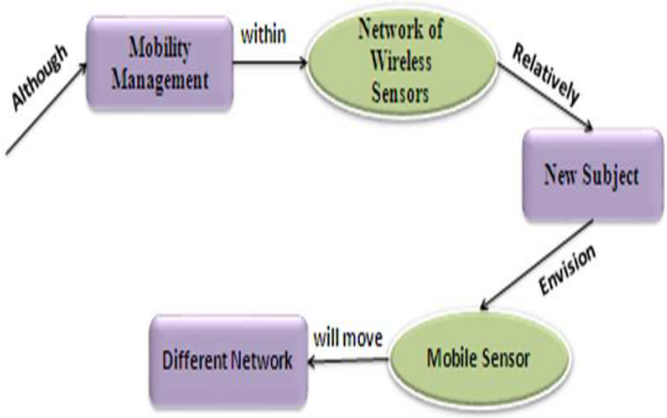

Fig.4 Data flow diagram : level 0

Here in this level diagram, although mobility management within a network of wireless sensors is a relatively new subject, we envision that a mobile sensor will move through different networks.

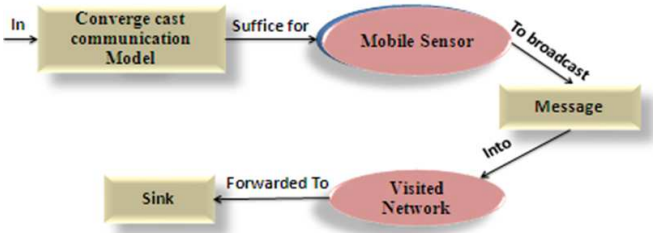

Fig.5 Data flow diagram : level 1

Here in this level diagram, In a converge cast communication model, it would suffice for mobile sensors to broadcast its messages into the visited network to have them forwarded to the sink.

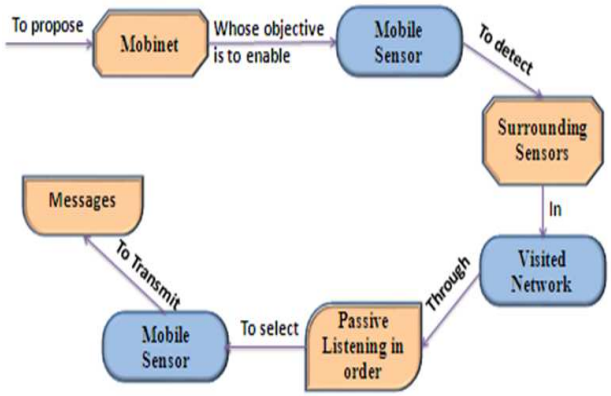

Fig.6 Data flow diagram : level 2

Here in this level diagram, we propose Mobinet whose objective is to enable mobile sensors to detect surrounding sensors in the visited network through passive listening in order to select one of them to transmit messages.

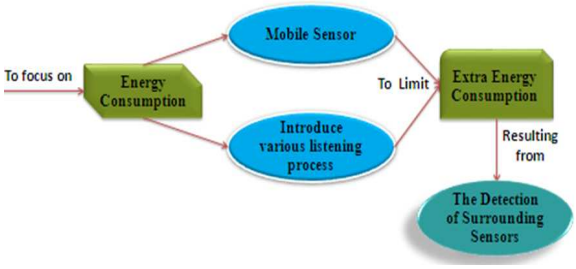

Fig.7 Data flow diagram : level 3

Here in this level diagram, we particularly focus on the energy consumption of the mobile sensors and consequently introduce various listening processes to limit the extra energy consumption resulting from the detection of surrounding sensors. Mobinet has been evaluated by simulation.

D. Context flow diagram:

A Context Diagram in software engineering and systems engineering are diagrams that represent all external entities that may interact with a system. This diagram is the highest level view of a system, similar to Block Diagram, showing a, possibly software-based, system as a whole and its inputs and outputs from/to external factors.

System Context Diagram are diagrams used in systems design to represent all external entities that may interact with a system. This diagram pictures the system at the center, with no details of its interior structure, surrounding by all its interacting systems, environment and activities. The objective of a system context diagram is to focus attention on external factors and events that should be considered in developing a complete set of system requirements and constrains.

System context diagram are related to Data Flow Diagram, and show the interactions between a system and other actors with which the system is designed to face. System context diagrams can be helpful in understanding the context in which the system will be part of software engineering. Context diagrams are used early in a project to get agreement on the scope under investigation. The contextual diagram of the overall project work is displayed below in Figure.

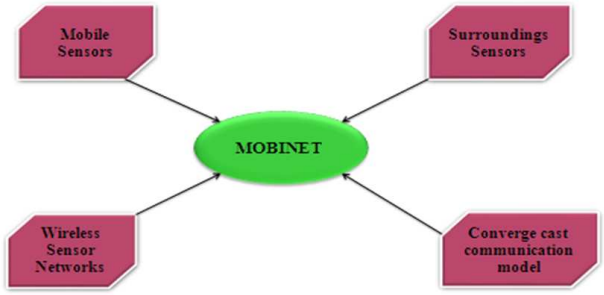

Fig.8 Context flow diagram

VII. DETAILED DESIGN

The introduction of structured programming in the 1960's and 70's brought with it the concept of Structured Flow Charts. In addition to a standard set of symbols, structured flow charts specify conventions for linking the symbols together into a complete flow chart. The structured programming paradigm evolved from the mathematically proven concept that all problems can be solved using only three types of control structures:

- Sequence
- Decision (Selection)
- Iterative (or looping).

A. Structured Chart

Structured flow chart gives overall strategy for structuring program. It gives details about each module evolve during detail design and coding. The modules and their design for this specific application is as shown in diagram. A structure chart is a top-down modular design tool, constructed of squares representing the different modules in the system, and lines that connect them. The lines represent the connection and or ownership between activities and subactivities as they are uses in organization charts.

A structure chart depicts

- the size and complexity of the system, and
- number of readily identifiable functions and modules within each function and
- Whether each identifiable function is a manageable entity or should be broken down into smaller components.

1) Structure chart construction

A structure chart can be developed starting with the creating of a structure, which places the root of an upside-down tree which forms the structure chart. The next step is to conceptualize the main sub-tasks that must be performed by the program to solve the problem. Next, the programmer focuses on each sub-task individually, and conceptualizes how each can be broken down into even smaller tasks. Eventually, the program is broken down to a point where the leaves of the tree represent simple methods

that can be coded with just a few program statements In practice, see figure, first is checked is a Structure Chart has been developed already.

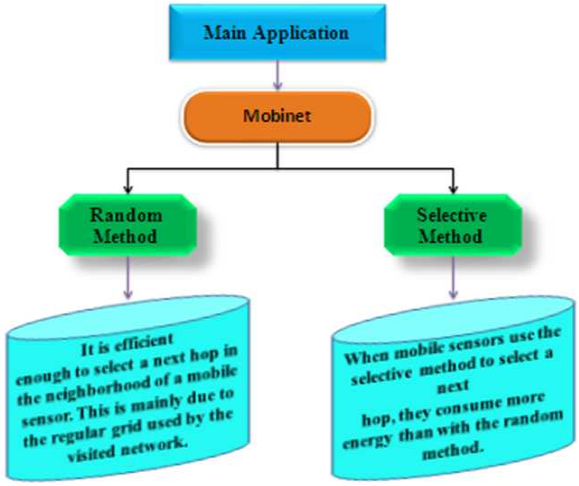

Fig.9 Structure chart

B. Sequence Diagram:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. A sequence diagram shows, as parallel vertical lines, different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

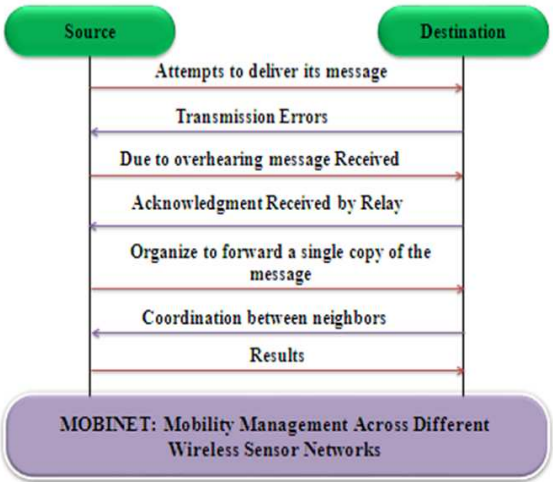

Fig.10 Sequence diagram

C. Process Flow Chart:

A flowchart is a common type of chart, which represents an algorithm or process, showing the steps as boxes of various kinds, and their order by connecting these with arrows. Flowcharts are used in analyzing, designing, documenting or managing a process or program in various fields.

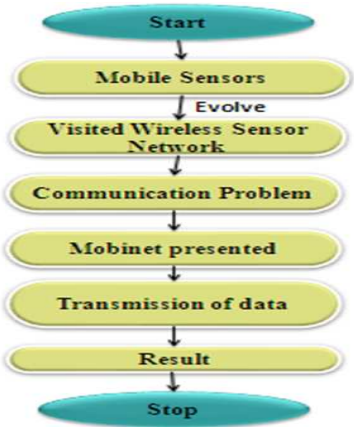

Fig.11 Process chart

VIII. CONCLUSION

In this article, we are interested in scenarios where mobile wireless sensors evolve in a visited wireless sensor network. We focus on the communication problems when the mobile sensors are moving in a visited network. We present a new approach named Mobinet which allows a mobile sensor to transmit data in the visited network. One of the advantages of Mobinet is that mobile sensors do not need to know which routing protocol is used in the visited network to forward messages toward the sink. Through neighborhood listening, our solution is able to select the next hop in the visited network for the messages. Mobinet supports two methods to select the next hop, the random method and the selective method, and multiple listening processes to reduce the energy consumption

REFERENCES

[1] [1] C.-Y. Wan, S. B. Eisenman, A. T. Campbell, and J. Crowcroft, “Siphon: overload traffic management using multi-radio virtual sinks in sensor networks,” in SenSys’05, 2005.
[2] G. Yang, B. Tong, D. Qiao, and W. Zhang, “Sensor-aided overlay deployment and relocation for vast-scale sensor networks,” INFOCOM’08, 2008.
[3] R. Kuntz, “Medium access control facing the dynamics of wireless sensor networks,” Ph.D. dissertation, University of Strasbourg.
[4] Q. H. Lilia Paradis, “A survey of fault management in wireless sensor,” Journal of Network and Systems Management, 2007.
[5] S. Biswas and R. Morris, “Exor: opportunistic multi-hop routing for wireless networks,” in SIGCOMM’05, 2005.
[6] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris, “A highthroughput path metric for multi-hop wireless routing,” in MobiCom’03, 2003.
[7] G. Schaefer, F. Ingelrest, and M. Vetterli, “Potentials of opportunistic routing in energy-constrained wireless sensor networks,” in EWSN’09, 2009.
[8] C.-J. Hsu, H.-I. Liu, and W. Seah, “Economy: a duplicate free opportunistic routing,” in Mobility’09, 2009.
[9] H. Frey, S. R¨uhrup, and I. Stojmenovic, “Routing in wireless sensor networks,” in Guide to Wireless Sensor Networks, 2009.
[10] A. Mei and J. Stefa, “Routing in outer space,” in INFOCOM’08, 2008.
[11] G. J. Pottie and W. J. Kaiser, “Wireless integrated network sensors,” Communications of the ACM, 2000.
[12] M. Buettner, G. V. Yee, E. Anderson, and R. Han, “X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks,” in SenSys’06, 2006.
[13] H. Karl and A. Willig, Protocols and Architectures for Wireless Sensor Networks. John Wiley & Sons, 2005.
[14] T. Camp, J. Boleng, and V. Davies, “A survey of mobility models for ad hoc network research,” WCMC’02, 2002. 356