Advanced Research in Electrical and Electronic Engineering
p-ISSN: 2349-5804; e-ISSN: 2349-5812 Volume 2, Issue 12 July-September (2015) pp. 23-28
© Krishi Sanskriti Publications
http://www.krishisanskriti.org/Publication.html

Role of Distributed Computing in Monitoring and Fault Tolerance of Reconfigurable Antenna Based Sensor Networks

Nilotpal Baruah[1] and Rocktotpal Baruah[2]

[1]Department of Computer Science Assam University, Silchar
[2]Department of Physics Tezpur University, Tezpur
E-mail: [1]nilotpaldu@gmail.com, [2]rocktotpalbaruah@gmail.com

Abstract—*Distributed Systems, with the ability to control and fault tolerance of most of modern remotely operated applications, starting from online transactions to ticket reservations, wireless sensor networks to Air Traffic Controls, are playing vital role in the field of networking and communication systems. Monitoring and controlling of wireless RF sensor networks comprising of reconfigurable antennas (RA) as sensors needs some fault tolerance mechanisms to overcome the faulty situations during sudden fault occurrence in the antennas. Utilization of reconfigurable antennas in RF sensor networks gives the dexterity to modify the network's objectives and coverage area, and cover-up faulty situations caused by unexpected break down of one or two antennas in the network. Thus a RA sensor network monitored and controlled by distributed computing will has the advantages of enhanced fault tolerance and agility in terms of objectives and coverage area. Here simulation based applications are can be used to monitor the process and bypass the faulty node in case of fault occurrence in the antenna without hampering the monitor process in the Distributed System environment.*

Keywords*: Distributed Systems, Reconfigurable Antenna, Fault, Fault Tolerance*

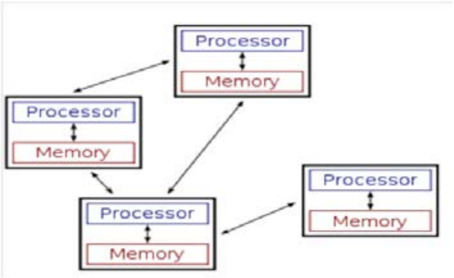
1. INTRODUCTION

The importance of Electronic sensor networks are increasing day by day in today's context in every aspect of life specially in public safety and security, health monitoring, data acquisition in process control as well as monitoring of different meteorological parameters which plays vital role in the field of aviation safety, precision agriculture etc. [1] which requires constant monitoring of the target area. Setting up just a sensor network cannot be taken as the solution of the problem, continuous monitoring and capability to overcome the unwanted situations, like fault occurrence in the system are must. RF sensor network can be used to set up a surveillance network using Reconfigurable Antenna (RA) systems. In this research paper we have presented an approach to find out how distributed systems can be used to monitor the sensor network even in the situation of fault occurrence.

2. DISTRIBUTED SYSTEM

A distributed system is a collection of independent computers that appears to its users as a single coherent system [2]. Distributed computing is a method of computer processing that studies distributed systems in which different parts of a program are run simultaneously on two or more computers that are communicating with each other over a network. In Distributed Systems, which actually a software system, components located over the network communicates over a method of message passing to complete their actions. In wider sense distributed system can be termed as an autonomous processes of interaction between various nodes over the same physical network by message passing. Basically a distributed system have the following properties:

i. It consists of several computers that do not share a memory or a clock.
ii. The computers communicate with each other by exchanging messages over a communication network.
iii. Each computer has its own memory and runs its own operating system.
iv. The resources owned and controlled by a computer are said to be local to it. The resources owned and controlled by other computers that can be only accessed through the network are said to be remote.



Source: [Wikipedia]

Fig. 1: Distributed System

2.1 Advantages Distributed System

i. Resource sharing: Sharing of hardware and software resources.

ii. Openness: Use of equipment and software from different vendors.

iii. Concurrency: Concurrent processing to enhance performance.

iv. Scalability: Increased throughput by adding new resources.

v. Fault tolerance: The ability to continue in operation after a fault has occurred

3. DISTRIBUTED SYSTEMS ARCHITECTURES

A Distributed system can be designed using two different models:

3.1. Client-server architectures

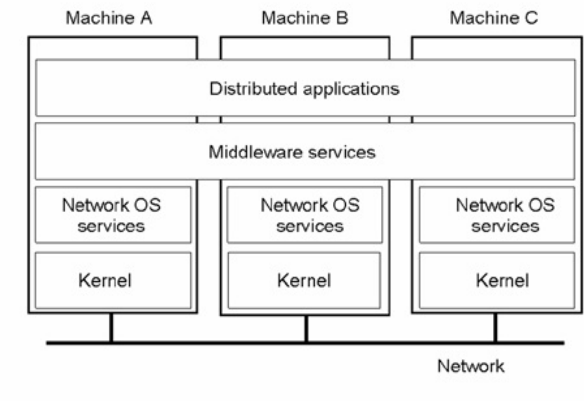
Distributed services which are called on by clients. Servers that provide services are treated differently from clients that use services.

3.2. Peer-to-Peer System

The term peer-to-peer is used to describe distributed systems in which labour is divided among all the components of the system. All the computers send and receive data, and they all contribute some processing power and memory to a distributed computation [3, 4].

4. MIDDLEWARE

Software that manages and supports the different components of a distributed system is termed as Middleware. In essence, it sits in the *middle* of the system. It is usually off-the-shelf rather than specially written software. Some examples are transaction processing monitors, data converters, communication controllers etc. [3, 4].



Source: [2]

Fig. 2: A Distributed System organized as Middleware

5. RECONFIGURABLE ANTENNAS BASED SENSOR NETWORK

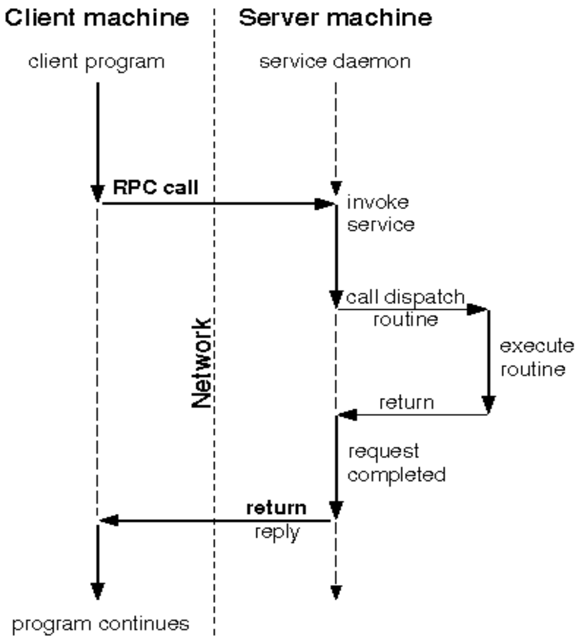
Reconfigurable Antennas (RA) are the antennas with capability of altering its operating parameters such as operating frequency, radiation pattern, polarization etc.[5] according to the utility. On the other hand, uses of antennas as RF sensors has been reported in various literatures [6]. Utilization of RA in RF sensors will be a promising one as RA will enhance the capability of sensor network by adding agility in terms of operating frequency and radiation pattern. The tunable operating frequency and radiation pattern gives adaptability to the sensor network in terms of objective and target area.

6. FAULT AND FAULT TOLERANCE USING DISTRIBUTED SYSTEMS

An ideal system is one which is perfectly reliable and never fails, which is impossible to achieve in practice. Systems fail for many reasons. A system can fail at several levels. Fault-tolerant design is a design that enables a system to continue its intended operation, possibly at a reduced level, rather than failing completely, when some part of the system fails. A distributed system has very important characteristics, i.e. fault tolerance. Fault-tolerant is a process where the system continues to perform correctly even when a subset of the processes becomes faulty. Fault-tolerance is highly desirable but often difficult to implement. A fault-tolerant system masks failures, i.e. continues to operate despite failures exhibits well defined failure behaviour, i.e. facilitates recovery but it may not continue operating [7, 8].

7. REMOTE PROCEDURE CALL (RPC)

Remote Procedure Call (RPC) is a powerful technique for constructing distributed, client/server based applications. RPC is called remote invocation or remote method invocation (RMI). An RPC is initiated by the client, which sends a request message to a known remote server to execute a specified procedure with supplied parameters. The remote server sends a response to the client, and the application continues its process. While the server is processing the call, the client is blocked (it waits until the server has finished processing before resuming execution), unless the client sends an asynchronous request to the server, such as an XHTTP call [9].



Source: [10]

Fig. 3: Remote Procedure Calling Mechanism

8. WORKING MECHANISM OF RPC

An RPC is analogous to a function call. Like a function call, when an RPC is made, the calling arguments are passed to the remote procedure and the caller waits for a response to be returned from the remote procedure. Fig. shows the flow of activity that takes place during an RPC call between two networked systems. The client makes a procedure call that sends a request to the server and waits. The thread is blocked from processing until either a reply is received, or it times out. When the request arrives, the server calls a dispatch routine that performs the requested service, and sends the reply to the client. After the RPC call is completed, the client program continues. RPC specifically supports network applications [9].

9. SYSTEMS MODEL AND OBJECTIVE

Similar to distributed system sensor network are also prone to fault occurrences like link failure between sensor nodes, sensor module crashes etc. Situations caused by such types of unpredicted fault occurrences can be overcome by using a network of sensor modules comprises of RAs as sensors and controlled by RPC protocol. In this literature a RPC controlled smart network of RA based sensors is presented. The network consist of a numbers of sensor modules located in a fixed geographical points. Each module has a processing unit, communication probe, sensing probe, storage unit and power unit. Modules sensing unit is an RA, which has the ability to change its operating parameters according to the utility will provide the required flexibility towards the fault tolerance. The fault tolerance problem in RF sensor network can be described using a network used for precision agriculture. The network deploys two different types of sensors for monitoring of insects in the crop fields (sensor A) and detection of bush fire surveillance (sensor B) and each will works on two different frequencies [11]. The networks target area layout is illustrated in the Fig. 4.

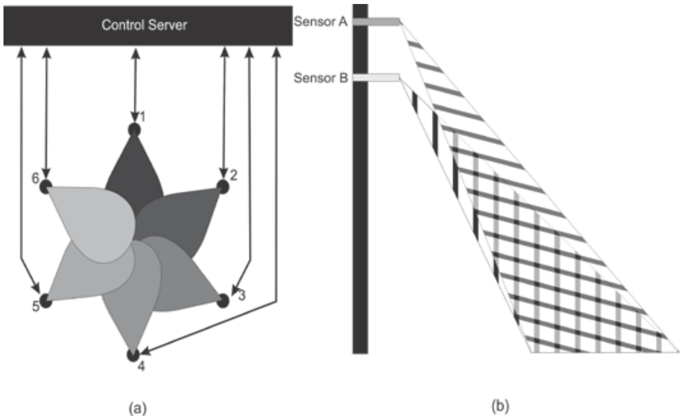


Fig. 4: Graphical illustration of the network

Fig. 1 (a) and (b) shows the top and side views of the sensor network and the shaded portions are the coverage zone of the sensors. In every positions (positions 1, 2, 3, 4, 5 and 6) both the types of sensors will be installed to monitor its assigned target area as shown in Fig. 1(b). All the sensors in the network will be interconnected through a central server system. In case of failure in any of the sensor module the central server can respond and it can assign sensor modules in the vicinity of the affected zone as backup measure.

In our study we are using different sensors based on Reconfigurable Antenna for the surveillance of crop field and are connected as distributed network. Each and every nodes are capable of communicate with the server with the help of message passing in the form of signals (RF/MW).

Our objectives are to monitor the system and fault tolerance during the situation of faulty node (sensor) occurs.

9.1 Case Study

The sensors in the network will communicate with the server through a beacon signal which will contain some critical parameters like module location, ID, task assigned, health of the module, remaining power level etc. in a predefined format on request or at a particular interval. Any discrepancy in the beacon signal from any sensor module will be treated as faulty scenario and the server will reassigned another sensor to that particular task. In the reassignment process priority of the task assigned will be a key parameter. Depending upon the task priority there may be two types of reassignments. The scenario can be illustrated using the Fig. 2.

Considering the task assigned to B type sensors are at higher priority and the sensor B at position become 2 faulty. As the sensor A at the particular position is in lower priority state server will reassigned to perform the high priority task by altering its operating frequency. Simultaneously the server will also readjust the cover zone of A type sensors in the position 1 and 3 by sweeping their radiation beam towards the faulty zone so that the window for the possible threat can be minimized. If the sensor A of position 2 will become faulty then the server will only readjust the radiation beam of the A type sensors of positions 1 and 3 and type B sensors are remain untouched they are already at higher priority.

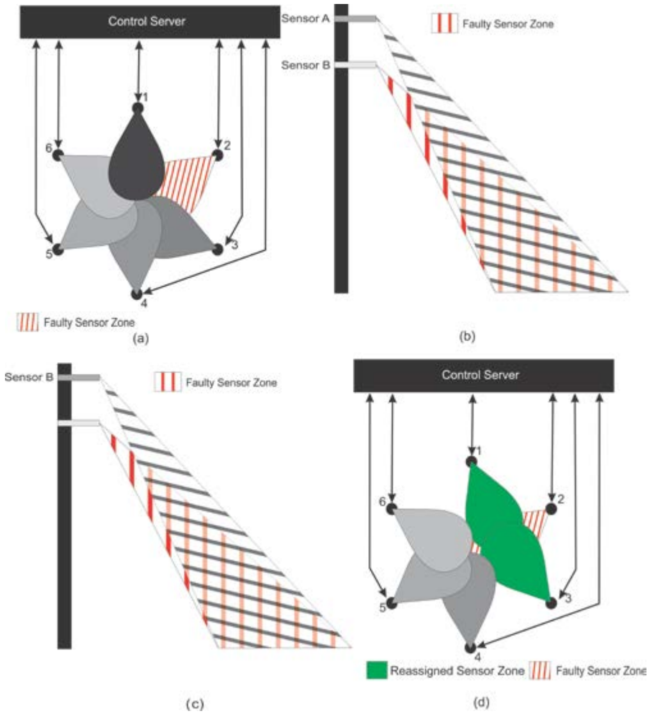


Fig. 5: Graphical illustration of fault occurrence and tolerance

10. FAULT DETECTION

Here we have used a Bayesian algorithm based fault detection method which was reported in [12]. To counter attack the problem of fault occurrence in sensor networks the system has to undergo two processes namely event detection and fault recovery. Event detection is related to detection of sensor functionality which is faulty. Generally an event should be detected as “event” by the sensors at the location but due to the faulty nature of the sensors an event may be termed as “no-event” or vice versa. Challenges lies in the disambiguation of an event from faults and this can be achieved by setting up threshold values or by examining the correlation in the readings of nearby sensors.

In real time situation the sensors can be following states

Table 1: Possible states of a particular sensor

S_D	B_D	Scenario
0	0	Correct normal reading
0	1	Faulty event reading
1	0	Faulty normal reading
1	1	Correct event reading

S_D = Sensor data; ‘0’ represents normal value ‘1’ represents unusual/ event related value

B_D = Beacon data; ‘0’ represents sensor is at normal region ‘1’ represent sensor in event region

10.1 Bayesian fault recognition algorithm

The sensor fault probability p is given by,

$p=Q((m_f - m_n)/2\sigma)$

Q is the Q-function

σ is the standard deviation

m_n is the mean of normal reading

m_f is the mean of event reading

Here it is considered that p is uncorrelated and non-symmetric i.e. $P(S_D=0|B_D=1)\neq P(S_D=1|B_D=0)$

From the fault probability equation it is derivable that the fault probability is low when mean normal and the event readings are not sufficiently distinguishable.

10.2 Decision scheme

Here, we have taken the threshold decision scheme. Consider the situation sensor i has N neighbors out of which k of them shows reading ‘a’ as i and other N-k shows reading ‘-a’, thus here evidence is $E_{i(a,k).}$ The decision for using the evidence can be taken by using the model

$P(R_i=a|E_i(a,k))= k/N$

R_i is the binary value with the decoded value

Using the following Bayesian calculation

$P_{aak}= P(R_i=a|S_i=b,E_i(a,k))= (p(N-k))/((1-p)k+p(N-k))$

P_{aak} is the conditional probability

The sensor makes a decision about whether or not to disregard its own sensor reading Si. If the P_{aak} value is less than certain threshold value sensor reading is taken as faulty otherwise reading is taken as correct.

11. SIMULATION

Simulation is the imitation of the operation of a real-world process or system over time [13]. Keeping in mind time, economical constraints as well as non-availability of actual experimental environment we have chosen simulation method for our study as it became more common, feasible and economically viable experimental tool in most of the experimental fields [14].

In our simulation model we have used one algorithm in the study based on the literature [15].

11.1 Algorithm

Step 1: Set up client server platform.

Step 2: Test condition using RPC

Step 3: Request message from server to all other clients which are connected to it.

Step 4: If clients are active they will reply by acknowledging the Request message with client Respond message.

Step 5: If all clients acknowledge to the server's then test condition succeeded or all clients are active.

Step 6: Server may request some signal from the sensor, sensor will process that request and send it back to the server.

Step 7: If any of the clients does not send respond to the server then that client is not active and that may sensor will lead to sink.

Step 8: When no response from the sensor the server will send a new request message from to a neighbor sensor of the sink client to overcome the fault occurrence.

As per the implementation of the algorithm for sensor networks we have taken five sensor nodes for consideration. We have done the simulation for two cases and results are given below.

We are taking the monitor process contains a token that contains two values i.e. '0' and '1'.The token revolve round the system in order to find alive and dead clients and returns back to the monitor process. The alive node is assigned by the Counter Value '1' and the dead node is assigned by the Counter Value '0'.Immediately after finding out the crash or abort clients, the monitor process sends messages to its most neighboring client to respond to get back the most possible signal.

In the first case all five nodes are active and working properly and sending response message to the server and is in a fault free environment.

Table 2: Status of different nodes (Case 1)

Nodes	Counter Value
A	1
B	1
C	1
D	1
E	1

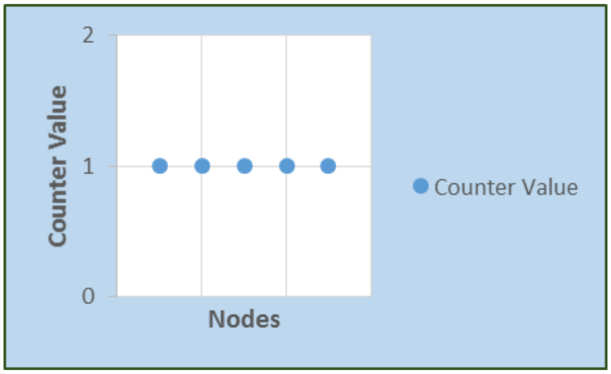


Fig. 6: Counter value against nodes in case 1

In the second case sensor node C is having some problem and not sending back ACK message to the server and lead to a faulty situation. Here the system overcomes as the most neighboring node D sends back the ACK message to the server, as the server redirects the

Request signal to the node D.

Table 3: Status of different nodes (Case 2)

Nodes	Counter Value
A	1
B	1
C	0
D	1
E	1

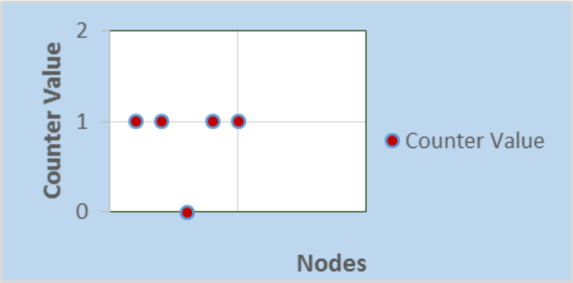


Fig. 7: Counter value against nodes in case 2

12. CONCLUSION

In our study we have used Bayesian fault recognition algorithm for detection of fault in the sensor network. The fault probability equation used in the study shows that the probability of fault is low in case of mean normal and event readings are not sufficiently distinguishable. We found that sensors are capable here to make decisions whether to disregard its own sensor readings or not.

Simulation results shows that the fault tolerance is possible in the model in the situations when faulty node occurs and signals can be retrieved from the nearest neighbor of the faulty node by the method of RPC.

Future works can be done using more efficient algorithms for better implementation of the fault free sensor networks.

The sensor makes a decision about whether or not to disregard its own sensor reading Si. If the P_{aak} value is less than certain threshold value sensor reading is taken as faulty otherwise reading is taken as correct.

REFERENCES

[1] G. Mao, B. Fidan, and B. D. O. Anderson, "Wireless sensor network localization techniques," *Computer Networks,* vol. 51, pp. 2529-2553, 7/11/ 2007.

[2] M. v. S. Andrew S. Tanenbaum, *Distributed Systems Principles and Paradigm*: Pearson Education, 2002.

[3] A. Bagchi and S. L. Hakimi, "An optimal algorithm for distributed system level diagnosis," in *Fault-Tolerant Computing, 1991. FTCS-21. Digest of Papers., Twenty-First International Symposium*, 1991, pp. 214-221.

[4] R. H. A.-D. Arpaci-Dusseau, Andrea C, *Introduction to Distributed Systems*: Arpaci-Dusseau Books, 2014.

[5] C. G. Christodoulou, Y. Tawk, S. A. Lane, and S. R. Erwin, "Reconfigurable antennas for wireless and space applications," *Proceedings of the IEEE,* vol. 100, pp. 2250-2261, 2012.

[6] S. Quegan, "Radar Remote Sensing," in *GEOINFORMATICS*. vol. 1, P. M. Atkinson, Ed., ed: UNESCO-Encyclopedia Life Support Systems, 2008, pp. 326-351.

[7] A. Avizienis, "Faulty-tolerant computing: an overview," *Computer,* pp. 5-8, 1971.

[8] J. G. Kuhl and S. M. Reddy, "Fault-diagnosis in fully distributed systems," in *Fault-Tolerant Computing, 1995, Highlights from Twenty-Five Years., Twenty-Fifth International Symposium on*, 1995, p. 306.

[9] N. B. J, "Remote procedure call," Ph.D., Computer Science, Carnegie-Mellon Univ., Pittsburgh, 1981.

[10] Y. Li, "Remote Procedure Call," ed, 2000.

[11] S. Bonafoni, F. Alimenti, G. Angelucci, and G. Tasselli, "Microwave radiometry imaging for forest fire detection: A simulation study," *Progress In Electromagnetics Research,* vol. 112, pp. 77-92, 2011.

[12] B. Krishnamachari and S. Iyengar, "Distributed Bayesian algorithms for fault-tolerant event region detection in wireless sensor networks," *Computers, IEEE Transactions on,* vol. 53, pp. 241-250, 2004.

[13] J. C. J. Banks, B. Nelson, D. Nicol *Discrete-Event System Simulation,* 5 ed.: Prentice Hall. , 2001.

[14] L. P. Saikia and K. Hemachandran, "Simulation of System Level Diagnosis in Distributed Arbitrary Network," *Journal of Theoretical and Applied Information Technology,* 2007.

[15] S. L. P. Kalita L., "A Study on Orphan Processes In the Distributed Systems Environments," *International Journal of Computer Science Engineering and Information Technology Research (IJCSEITR),* vol. 4, pp. 47-54, 2014.