NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

JOANA FALCÃO DE MOURA

BsC in Computer Science

# IOC: INTERNET OF COWS

TOO BE ADDED

Dissertation Plan
MASTER IN COMPUTER SCIENCE AND ENGINEERING

NOVA University Lisbon
***Draft:*** *January 19, 2023*

DEPARTMENT OF
COMPUTER SCIENCE

# IOC: INTERNET OF COWS

TOO BE ADDED

**JOANA FALCÃO DE MOURA**

BsC in Computer Science

**Adviser:** João Carlos Antunes Leitão
*Assistant Professor, NOVA University Lisbon*

**Co-adviser:** João Nuno de Oliveira e Silva
*Assistant Professor, University Lisbon*

Dissertation Plan
MASTER IN COMPUTER SCIENCE AND ENGINEERING

NOVA University Lisbon
***Draft:*** *January 19, 2023*

# Abstract

# Resumo

# Contents

# List of Figures

# List of Tables

# Acronyms

| | |
|---|---|
| **CREW** | Concurrent Random Expanding Walkers *(p. 6)* |
| **GPS** | Global Positioning System *(p. 10)* |
| **NeEM** | Network Friendly Epidemic Multicast *(p. 5)* |
| **Scamp** | Scalable Membership Protocol *(p. 5)* |
| **TCP** | Transmission Control Protocol *(p. 5)* |
| **WSN** | Wireless Sensor Networks *(p. 7)* |

# 1

# Introduction

This chapter will explore the motivations for this thesis development, the underlying problem that provoced it, the objectives expected to achive during its development and finish with the current document structure.

## 1.1 Motivations

In the xxx Farm, located in xxx, they have over 150 cows, alongside many other animals, spread throughout xxx acres. Controling that many animals in such a vaste terrain is quite a difficult task. Futhermore, the region lacks cellular service, which complicates this mission even more.

The cows are kept seperated in herds depending on their ages, this means that the younger cows are not in the same herd as the older ones. This kind of seperation is quite important for them to coexist. Inside each herd they follow a hierarchical structure, having a leader that all the other cows follow.

The xxx Farm currently has physical fences in place to mantain the multiple herds seperated from each other and protected. However, this fences are not very persistent, which lead to an often replacement and potencial danger for the cows. In addition, since the farm has an immensive amount of land, it is reasonably strenuous to locate all cows and make sure all are healthy and safe.

Having already available some great options of collars that create virtual fences for all kinds of animals, there are still no alternative that would work for the xxx Farm. Mainly because of the lack of cellular service available, but also do to the immensely amount of cattle at this location.

## 1.2 Problem Statement

Currently there is no low consuming, reliable and robust to faults collar for tracking cattle, with the option to create a virtual fence for the herds, change its position and ultimatly change the cows course when they deviate from their path.

## 1.3 Objectives

During this dissertation it is expected to be developed a fully functional animal collar, adaptable to any cow, that connects to gps and creates a virtual fence for each herd. This fence should be adjustable accordingly to the user's desire and the collars should send a vibration to the cows, if they are located ouside the fence area, in order to get them back inside.

These collars should provide accurate information about the cows locations as well as be highly scalable to handle all the cows informations. The ultimate goal is to find a reliable and not much consuming solution to deal with the sensing informations.

## 1.4 Structure

The remainder of this dissertation is organized as follows:

- Chapter 2 - Related Work: includes research on existing protocols for broadcasting, focusing on Gossip Protocol, options for wireless sensor networks, available sensors and Arduino alternatives, how cows behave in a herd and lastly some possible existing collars.
- Chapter 3 - Work Plan: a description of the future work organization and explication of each work phase.

# 2 Related Work

## 2.1 Gossip Protocol

### 2.1.1 History and Overview

The gossip protocol, also known as epidemic protocol, as the name indicates, was created based on how gossips are propagated in social groups. In a gossip protocol, nodes in a network send the information, randomly, to other nodes in the same network, similar to how a gossip is spread between members in a social group [9].

The gossip protocol is known as a highly scalable and resilient approach to implement reliable broadcast. This protocol is based on every participant propagating their messages collaboratively with all the members of their group.

This process starts when a node desires to propagate some piece of information to the other members of his network. This node will send his message to $t$ nodes, chosen randomly, ($t$ being a parameter called *fanout*, which is better explaned in the subsection 2.1.2). When the receiving nodes obtain the message for the first time, they will do the same as the previous node had done and resend the message to $t$, randomly chosen nodes. If a node receives the same message twice, it will discard it. When this happens, which may occour quite often, since the nodes are unaware of which nodes have already received a message, there is redundancy.

However, since neither node knows who has received each message and who has sent a message to whom, each node will have to keep a log of all messages that he has already received.

### 2.1.2 Parameters

**Fanout:**

**Maximum Rounds:**

### 2.1.3 Strategies

The Gossip Protocol may be executed following different approaches [8]:

**Eager push approach:** As soon as a node receive a message for the first time, it sends it to *t* randomly selected nodes. This approach consumes a great amount of bandwidth, considering it leads to multiple copies of the same messages are delivered to each target node.

**Pull approach:** Regularly, nodes inquire each other on new messages they've recently receive. If they adquire information about a message they haven't receive yet, they will request it expecifically from that node. This approach leads to higher values of latency, derivated from the extra round trip needed to obtain the messages.

**Lazy push approach:** When a node receive a message for the first time, it will only broadcast to its neighbours a small fraction of the message, a identifier, per example a hash of the message. If the neighbour never receive the given identifier it will request the rest of the message. As in the pull approach, there will be a higher value of latency.

Besides the diferences in latency and bandwidth previously mentioned, there is another important distinction between the eager push approach and the pull and lazy push approaches. Considering that the eager push approach sends the entirety of each message immediatly after receiving it, the nodes do not need to manitain a copy of these messages, contrarly to the other two approaches that may need to resend these messages later. This leads to a higher memory requirement for these approaches [10].

By combining the approaches studied above, we can get better results, obtaining a better latency/bandwidth tradeof. This are two of the studied combined approaches [4]:

**Eager push and pull approach:** This method is divided between two distinct phases. The first phase consists on using the eager push approach to disseminate messages straightly to the nodes in the network. The second phase uses the pull approach to recover the lacunas that might have occoured during the first phase of this method. This approach reduces the amount of redundancy in comparinse with the eager push approach, without decreasing its performance. It will, however, endure a higher level of latency due to the pull phase.

**Eager push and lazy push approach:** It is used the eager push approach to a subset of nodes. Then it uses the lazy push approach on the remaing subset of nodes to recover the lacunas and guarantee the reliable of the method.

### 2.1.4 Tree-based Approaches

Tree-based broadcasting methods have a small message complexity, however, they are not particularly resilient to faults. On the other hand, gossip protocols, as mentioned earlier in subsection 2.1.1, are known for their resilience, but have a high message complexity [11].

In order to obtain a small message complexity and high reliability, it was considered combining both these methods.

With this protocol we obtain the nodes organized in a tree structure formate, where each node knows to whom foward its messages. To achieve this structure we have many approaches, per example the PlumTree protocols:

**PlumTree protocol** This protocol uses eager push and lazy push gossip, previously explaned in the subsection 2.1.3. It separates the nodes in the network in two subsets of randomly selected nodes. The first subset of nodes uses the eager push protocol to disseminate the messages, while the other uses the lazy push protocol. The links that the eager push method uses to propagate the messages are chosen to create a randomized broadcast network using a tree-based structure. While the links used during the lazy push gossip are used to ensure the reliability of the method when nodes fail and potencially heal the broadcast tree when needed [11].

Additionally, in opposition to other gossip protocols, with tree-based gossip the connections first made by the eager push propagating will remain until it is detected a failure. This will allow us to use TCP connections, which will provide extra reliability and failure detection.

### 2.1.5 Examples

Throughout many years there have been proposed numerous gossip-based protocols. During this section we will discuss some of them:

**Scalable Membership Protocol (Scamp):** Contrarly to many other gossip-based protocols, with scamp it is proposed that the individual nodes have a randomized partial view of the global members in the network, leading to a fully descentralized system. This is quite an important advantage for large scale groups, since it requires a significant amount of memory and generates a lot of network traffic to maintain the system's overall consistency in extensive groups. Additionally, the scalable membership protocol is also compelling for its natural increase and reorganization of the partial view of the system when new nodes are added to the network. Having this partial view around *log n* nodes (being *n* the number of overall nodes in the network) [6].

**Network Friendly Epidemic Multicast (NeEM):** One of the biggest problems in most gossip-based protocols is when the network gets congested and, subsequentially, the messages get lost. NeEM uses Transmission Control Protocol (TCP) to disseminate the messages and resolve this problem, with the usage of its ineherant flow and congestion control mechanisms. In order to maintain the protocol's stability, NeEM uses a buffer management technique that utilizes different approaches to discard messages on overflow. It also includes the knowledge about the messages' types in order to ensure that the buffer retains enough space and bandwidth is used to better fit each request [13].

**Concurrent Random Expanding Walkers (CREW):** Crew is a gossip-based protocol designed to minimize the messages dissemination speed. This is acquired by maintaining in cache the information about the already established connections, which will reduce the latency of reopening a TCP connection [5].

**Scribe:** Scribe is a large scale and fully descentralized

### 2.1.6 Gossip Limitations

Throughout this section it has been vastly mentioned the advantages provided by the gossip protocol. Mainly, it was refered the resilience and scalability offered by this method. However, as any other protocol, it has its limitations. A few of this are [3]:

1. Fixed maximum message size - the gossip protocol has a fixed maximum message size which may lead to problems. Per exemple, if we desire to propagate a message with a greater size than this fixed maximum message size, we will have to divide the message into multiple messages. When this happens, some of these messages may be lost, since each node can only gossip a certain amount of information per round, which will increase the number of rounds to deliver a single message.

2. Slow rate - the rate of messages exchange in a gossip protocol is typically quite slow, which can cause some complications when handling sudden events. This situation might be surpassed by reducing the periodicity of messages exchanges, however, this often leads to yet another problem, the increasing of overheads.

3. Malicious behaviours and correlated loss patterns - another gossip protocol's limitation is when the nodes behave in a malicious form, intentionally, with disseminating of false information or when nodes malfunction, or unitentionally, when multiple nodes fail or become unavailable at the same time.

### 2.1.7 Discussion

Throughout this section 2.1 it has been explained the basis of the gossip protocol, its strategies, tree-based approaches, particularly the PlumTree, it was also mentioned some interesting examples of gossip-based methods and finally its limitations.

This protocol is crutial for the development of my dissertation, since it is highly scalable and reliable, which is fundamental for dealing with the messages exchanges between all cows. Every cow will have to be constantly informed on the fence location, since it can potencially change. Using the gossip protocol to disseminate this information seems benefic.

The PlumTree approach is quite interesting for my dilema considering that every herd has a leader that commands all the other cows in the group. With a tree-based strategy this cow appears to be an obvious choice to be the root and send the information throughout the entire herd.

## 2.2 Wireless Sensor Networks

### 2.2.1 Definition

Wireless Sensor Networks (WSN) is an innovative technology with immeasurable applications ranging from remote environmental monotorization to target tracking. This network is composed by multiple small cheap and low-power sensor nodes distributed throughout various locations. Usually, these nodes are scattered in a sensor field, as demonstrated in Fig. 2.1. Individually, each node can perform sensing tasks, which implies:

- collecting data, per example, from its surroundings, such as temperature, light, humidity, and many other types of data, depending on its sensor;
- process it, using its on-board processor;
- and finally, transmite it back to the sink by multi-hopping. It eventually reaches the end users via internet or a satellite [2].

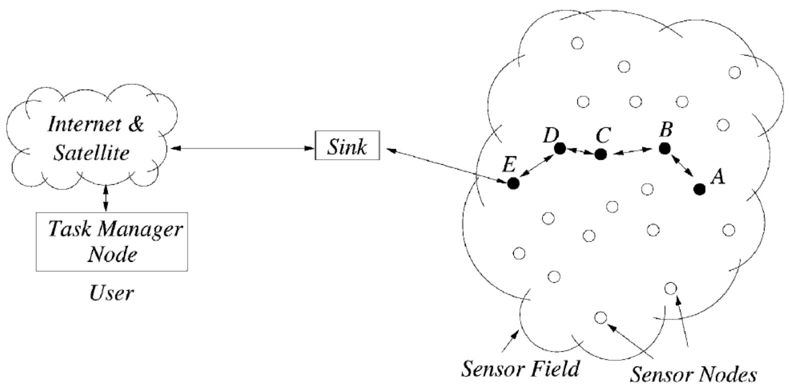

Figure 2.1: Sensor nodes in a sensor field [2]

Typically, each node sensor is composed by a radio trasceiver, an embedded processor, internal and external memories, a power source and one or more sensors [14]. However, the nodes' design and the WSN itself' must take into account its purpose, the environment where the nodes will operate, the hardware, the cost, along with other restrictions that need to be considered [16].

### 2.2.2 Architecture

The most common architecture for WSN follows the OSI model. This model is composed of five layers: application layer, transport layer, network layer, data link layer and physical layer, and three cross plane layers: power management plane, mobility management plane and task management plane, as shown in Fig. 2.2.

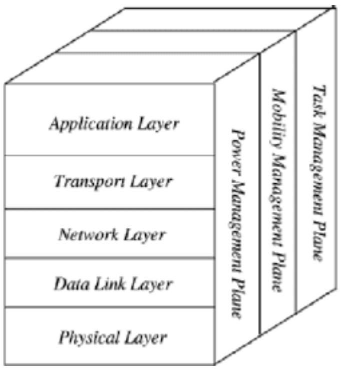

Figure 2.2: WSN Architecture [2]

The five layers above mentioned work together to ensure the data is properly transmited to the network, each with a specific functionality:

**Application Layer**

**Transport Layer**

**Network Layer**

**Data Link Layer**

**Physical Layer**

The management planes' roles are to manage the network and optimize the sensor nodes' performance in order to improve the overall effectiveness of the network, considering the advantages acquired by all sensor nodes working together. Each of these planes manages a specific area [2]:

**Power Management Plane** manages how the sensor node uses its power, chosing when to turn off its receiver to save energy or to keep it from receiving repeted messages. It also informes its neighbours when it reaches a low power mode.

**Mobility Management Plane:** keeps track of the sensor nodes' neighbours and always disntinguishes a route back to the user.

**Task Management Plane:** administers the periodicity and schedule that each node needs to maintain in order to perform their sensing tasks based on their power dependency and task requirements.

### 2.2.3 Network Topologies

There are several different topologies regarding the connection between nodes and their message exchange routes [15, 12]:

**Star Topology:** all nodes are connected to only one node, the coordinator. This means every node will communicate via this central node and every node that requests to enter this network will have to send its information to the coordinator, which will then send it to the other nodes. The principal limitation of this topology is that if the coordinator malfunctions the whole network will fail.

**Ring Topology:** all nodes are equal connected, having no coordinator. Contrarly to the star topology, if a single link is broken the whole network will fail.

**Bus Topology:** all nodes broadcast their messages using the bus. Each message has a header with the destination address so that every node can see if the message is fot them or another node. This topology is passive, since the nodes are not responsible for retransmitting messages.

**Tree Topology:**

**Fully Connected Topology:** every node is connected to every other node. This will lead to a routing problem when dealing with large networks.

**Mesh Topology:** the nodes are generally identical, so the mesh connections are commonly referred as peer-to-peer connections. However, even though the nodes are generally identical some of them can be assign as coordinators that take additional functions and if one of these coordinators stops working, another just takes over his work. An interesting aspect of this topology is that the communicattion can be done between any two nodes in close proximity, which makes this topology quite robust to the failure of nodes or links and good for large scale networks.

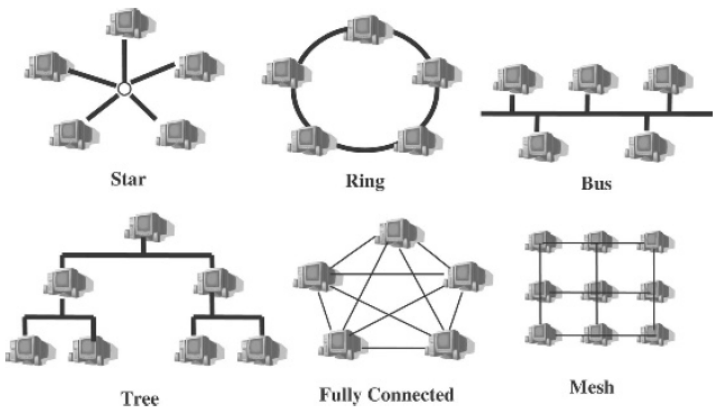

Figure 2.3: Network Topologies [12]

### 2.2.4 Strategies

The WSN have enumerous applications, this idea will be further discussed in the subsequent subsection 2.2.6, which will lead to different solutions for each of them.

### 2.2.5 Gossip in WSNs

One of the main purposes of a sensor node is to transmite the data it has collected, via the sensors, to the sink. The route chosen by these nodes is of the most importance, therefore various protocols were studied in [1] to understant which of these would better conduct this task.

As presented previously in the section 2.1, during the gossip protocol each node only transmites its messages to *t* randomly selected nodes and not the whole network, as in the flooding protocol. This characteristic ensures that every node in the gossip protocol will only have a single copy of the packet to be sent, which adresses one of the shortcomings of the flooding protocol, the implosion. However, this will lead to delays in the dissemination of the data which may be an important factor for some applications of the network.

### 2.2.6 Applications

The WSN has a vast number and types of applications, some are related to the health, others to the military, home, environmental and many others.

#### 2.2.6.1 ZebraNet

One of the most revolucionary application of WSN was developed to track wildlife, specifically zebras, for biology research, using a mobile base station. This method, denominated ZebraNet, collects logged data from tracking collars, transported by the animals, and afterward it transmites this data back to the researchers. Considering there is no fixed antennas or cellular telephone service, the protocol uses ad hoc peer-to-peer routing to transport the data around.

The ZebraNet project focused on resolving some of the problems observed from previous studies of collecting data from wildlife tracking. One of the main obstacle was using satellites to transport the data. The process of uploading data to satellites is slow and power consuming. Moreover, the data download from the satellite to the researchers is charged by the bit, which restricted the amount of data collected. Futhermore, these systems used bateries without solar recharge, which would eventually end, and had to be recovered and recharged, losing enormous amounts of data in process [7].

Additional, one of the biggest concerns during the development of this project was the design limitations. Due to the fact that each node would be transported by an animal, its weight and size was immediatly limited. And since the nodes are difficult to retrieve, the device has to have a durable battery life [17]. Subsequentially, most of the weight would be occupied by the battery and the Global Positioning System (GPS), leaving a small space for the storage, which means that there is small space for redundant messages in this protocol. Lastly, it was crucial to consider the impact of the number and size of data transmissions required as well as the range of these transmissions.

#### 2.2.6.2 Wireless Tracking

### 2.2.7 Discussion

## 2.3 Existing Collars

## 2.4 Cows Habits

## 2.5 Summary

# 3 Work Plan

# Bibliography

[1] K. Akkaya and M. Younis. "A survey on routing protocols for wireless sensor networks". In: *Ad hoc networks* 3.3 (2005), pp. 325–349 (cit. on p. 10).

[2] I. F. Akyildiz et al. "Wireless sensor networks: a survey". In: *Computer networks* 38.4 (2002), pp. 393–422 (cit. on pp. 7, 8).

[3] K. Birman. "The promise, and limitations, of gossip protocols". In: *ACM SIGOPS Operating Systems Review* 41.5 (2007), pp. 8–13 (cit. on p. 6).

[4] N. Carvalho et al. "Emergent structure in unstructured epidemic multicast". In: *37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'07)*. IEEE. 2007, pp. 481–490 (cit. on p. 4).

[5] M. Deshpande et al. "Crew: A gossip-based flash-dissemination system". In: *26th IEEE International Conference on Distributed Computing Systems (ICDCS'06)*. IEEE. 2006, pp. 45–45 (cit. on p. 6).

[6] A. J. Ganesh, A.-M. Kermarrec, and L. Massoulié. "Scamp: Peer-to-peer lightweight membership service for large-scale group communication". In: *International Workshop on Networked Group Communication*. Springer. 2001, pp. 44–55 (cit. on p. 5).

[7] P. Juang et al. "Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with ZebraNet". In: *Proceedings of the 10th international conference on Architectural support for programming languages and operating systems*. 2002, pp. 96–107 (cit. on p. 10).

[8] R. Karp et al. "Randomized rumor spreading". In: *Proceedings 41st Annual Symposium on Foundations of Computer Science*. IEEE. 2000, pp. 565–574 (cit. on p. 3).

[9] J. Leitao. "Gossip-based broadcast protocols". MA thesis. Master's thesis, University of Lisbon, 2007 (cit. on p. 3).

[10] J. Leitao. "Topology Management for Unstructured Overlay Networks". In: *Technical University of Lisbon* (2012) (cit. on p. 4).

[11] J. Leitao, J. Pereira, and L. Rodrigues. "Epidemic broadcast trees". In: *2007 26th IEEE International Symposium on Reliable Distributed Systems (SRDS 2007)*. IEEE. 2007, pp. 301–310 (cit. on pp. 4, 5).

[12] F. L. Lewis. "Wireless sensor networks". In: *Smart environments: technologies, protocols, and applications* (2004), pp. 11–46 (cit. on p. 9).

[13] J. Pereira et al. "Neem: Network-friendly epidemic multicast". In: *22nd International Symposium on Reliable Distributed Systems, 2003. Proceedings.* IEEE. 2003, pp. 15–24 (cit. on p. 6).

[14] Q. Wang and I. Balasingham. "Wireless sensor networks-an introduction". In: *Wireless sensor networks: application-centric design* (2010), pp. 1–14 (cit. on p. 7).

[15] S. Yadav and A. Chitra. "Wireless sensor networks-architectures, protocols, simulators and applications: a survey". In: *International Journal of Electronics and Computer Science Engineering* 1.4 (2012), pp. 1941–1953 (cit. on p. 9).

[16] J. Yick, B. Mukherjee, and D. Ghosal. "Wireless sensor network survey". In: *Computer networks* 52.12 (2008), pp. 2292–2330 (cit. on p. 7).

[17] P. Zhang et al. "Hardware design experiences in ZebraNet". In: *Proceedings of the 2nd international conference on Embedded networked sensor systems*. 2004, pp. 227–238 (cit. on p. 10).