RFIDPlanner - A Coverage Planning Tool for RFID Networks

Nitesh Gupta
Dept. of Computer Science and Engineering
Indian Institute of Technology - Bombay
Mumbai, India
Email: nitesh@cse.iitb.ac.in

Sridhar Iyer
Kanwal Rekhi School of Information Technology
Indian Institute of Technology - Bombay
Mumbai, India
Email: sri@it.iitb.ac.in

***Abstract*— Radio Frequency Identification or RFID systems are emerging as economical solutions for fast identification of objects. One of the important aspects of setting up a RFID network is deploying RFID readers to ensure complete coverage of RFID tags in a given area. This task is usually accomplished by conducting site surveys and then deploying RFID readers in different configurations to determine the optimal one. As the size of the given area increases, time taken to setup a RFID network increases thus increasing the cost of deployment.**

We present *RFIDPlanner*, a coverage planning tool for RFID networks, which attempts to tackle the issue using simulation. Given the details of an area, the properties of obstacles present in the area, and the specifications of the RFID reader and tags used, *RFIDPlanner* simulates the coverage pattern of a reader giving a graphical view of the RFID reader's interrogation range. Through its interactive GUI the layout can be modified during run-time, by moving and deleting existing RFID readers and adding new ones. The zoom feature allows different levels of visualizations ranging from a view of the entire layout to focused views of particular sections and RFID readers.

I. Introduction

A Radio Frequency Identification (RFID) System consists of RFID Tags and RFID Readers at its core, performing the main task of identifying individual objects using radio waves. An RFID Reader consists of a radio frequency module for two way communication with tags along with a control unit. It interrogates the tags or reads the information stored in the tags by transmitting radio frequencies which are reflected back by the tags along with the data.

An RFID Tag is a microchip coupled with a small antenna and is tagged to the object to be identified. The microchip carries the data associated with the object it is tagged to, usually a serial number also known as the Tag ID. The tag responds to the reader interrogation by modulating its data on the interrogation signal and using it for communication with the reader. RFID Tags can be classified as active and passive. Active tags are equipped with onboard power supply and hence have better processing abilities and a longer transmission range. On the other hand, passive tags derive power from the interrogation signals broadcasted by the reader, and thus have limited processing ability and a smaller transmission range. To remain cost-effective, most of the RFID networks use passive tags. Hence, for the design of *RFIDPlanner* we consider only passive RFID tags.

RFID technology promises to make a big impact on data intensive business processes such as supply chain, stock management and warehousing which require rapid identification of a huge number of objects. RFID technology provides a fast mechanism for object identification, increasing efficiency with lower error rate. Other applications of this technology include equipment and livestock tracking as well as security purposes, such as tracking of and theft prevention in automobiles.

Since RFID readers have small interrogation ranges, deploying an RFID network involves large number of RFID readers. To be cost-effective, it becomes important to deploy minimum number of readers while ensuring complete coverage of the area where RFID tags might be present. The procedure to determine such an optimum configuration typically involves site surveys by surveyors who have extensive knowledge of radio wave characteristics and behaviors patterns in presence of different materials. The surveys involve testing different positioning of readers and tracking reader's range patterns, which can change with a slight change in the readers position or a new obstacle in the reader's range. The entire deployment procedure takes up significant time and increases deployment costs. Thus, deploying RFID readers in a fast and cost effective way is a difficult problem and requires automated tools designed specifically for this purpose.

The problem of coverage planning in RFID networks can be thought of as similar to coverage planning in indoor WLANs. Just like RFID readers in RFID networks, access points in WLANs need to be placed optimally to ensure complete coverage of a given area taking into account the different obstacles present in the area. Several simulation tools have been developed to aid coverage planning using propagation modeling. Some examples of such tools are WISE developed at AT&T [6] and SpectraGuard Planner developed by AirTight Networks [7].

However, it is not easy to adapt WLAN coverage tools for RFID networks since there remain some basic differences between the two. To remain cost-effective RFID networks are constrained to use passive RFID tags that depend upon the RFID reader for theri power supply. Hence there is a limit on the interrogation range of a RFID reader. On the other hand an access point interacts with wireless devices such as laptops, PDA, etc. which have superior processing capabilities, higher tolerance to communication errors and a power supply of their own. These devices are thus capable of communicating with access points over relatively larger distances. These differences make it difficult to use tools designed for WLANs for coverage planing in RFID networks. Therefore, there is a need to build specific coverage tools for the RFID domain. To the best of our knowledge no work has been done specifically on coverage tools for RFID systems.

Through this paper we wish to discuss the various design parameters which should be taken into consideration while designing a coverage planning tool for RFID networks, suggest an architecture for the same and present some algorithms which can be useful in emulating range patterns and automatically generating RFID reader layout configurations.

We have designed and developed *RFIDPlanner* - a coverage planning tool. A detailed description of the design and architecture is presented in Section II. The main assumptions, design drivers, formulae for calculating range and the propagation model used are given in Sections II-A, II-B, II-C and II-D. *RFIDPlanner* can emulate a reader's range attenuation patterns given the details of the area to be covered, properties of the obstacles present in the area, and reader and tag specifications. The algorithm for emulating reader range is explained in Section II-E. *RFIDPlanner* provides an interactive layout visualization and modification unit which can be used to visualize the range attenuation patterns in different locations and modify reader layout configurations during runtime. This is also highlighted in Section II-E. A case study of a shopping mall scenario is presented in Section III. Two heuristics designed to automatically generate reader layout configurations are discussed in Sections III-A and III-B.

II. *RFIDPlanner*

In this section we decribe the assumptions, design drivers, antenna range calculations, propagation modeling and architecture involved in the design of *RFIDPlanner*.

A. Assumptions

Some of the assumptions made while designing *RFIDPlanner* are stated below.

- RFID readers being used are of the same type, having same characteristics and hence the same maximum interrogation range.
- RFID tags being used are passive and of the same type.
- Antennas are isotropic having circular ranges.
- Obstacles are rectangular and are either aligned along the x-axis or along the y-axis.

B. Design Drivers

RFIDPlanner has been designed keeping in mind the following objectives.

- Place readers statically to cover the given area
- Simulate the interrogation range attenuation patterns for RFID readers in presence of different obstacles.
- Provide a visualization tool that displays the layout of readers in the given area.
- Allow users to modify the layout during run-time allowing them to move and delete existing RFID readers and add new ones.

C. Antenna Range

The interrogation range for an antenna, in absence of any obstacles, is calculated on the basis of power supplied by its parent reader. The power received at a distance r from the source, i.e. the electromagnetic coupled antenna, is inversely proportional to the square of that distance r^2 [1]. For a given power the interrogation range of an antenna is defined as the maximum distance r at which the received power is just sufficient to make the tag operational, i.e., the tag is able to process the RF signal and reflect back information. This interrogation range r can be calculated using the following formula [1]

$$r = \sqrt{\frac{P_{RI} * 10^{(G_R/10)} * 10^{14.76} * G_T * G_R}{(P_{TI})_{min} * f^2}} \tag{1}$$

where f is the transmission frequency, G_T is transponder's antenna gain (in dBd), G_R is reader's antenna gain (in dBi), P_{RI} is power input at reader's antenna and $P_{TI})_{min}$ is minimum power input required at tag's antenna to make it operational.

Even though the interrogation range of the reader's antenna can be increased by increasing the power level, a maximum is reached when the power reflected back by a just operational tag is just sufficient for the reader's antenna to detect the tag. On increasing the input power of reader's antenna a tag at a further distance might become operational but its response would not be detected by the reader's antenna. This maximum possible range can be calculated in the same fashion as equation (1)

$$r_{max} = \sqrt{\frac{(P_{TO})_{min} * 10^{14.76} * G_R * G_T}{(P_{RI})_{min} * f^2}} \tag{2}$$

where $(P_{TO})_{min}$ is minimum power output by a just operational tag and $(P_{RI})_{min}$ is minimum reflected power from tag required as input at reader's antenna for it to detect the tag

D. Propagation Modeling

Propagation models can be classified as deterministic and empirical [2]. Deterministic modeling is based on electromagnetic wave propagation theory and makes use of ray tracing techniques to model range patterns. It requires detailed description of the area under consideration which can often be complicated and difficult to obtain.

On the other hand, Empirical modeling makes use of statistically processed representative measurements of propagation losses due to different obstacles that lie in the path of the electromagnetic waves. Empirical modeling assumes that effects of reflection, diffraction and scattering on the range patterns are taken care of in the representative measurements. The propagation loss between a transmitter and a receiver L [2] is given by the following equation

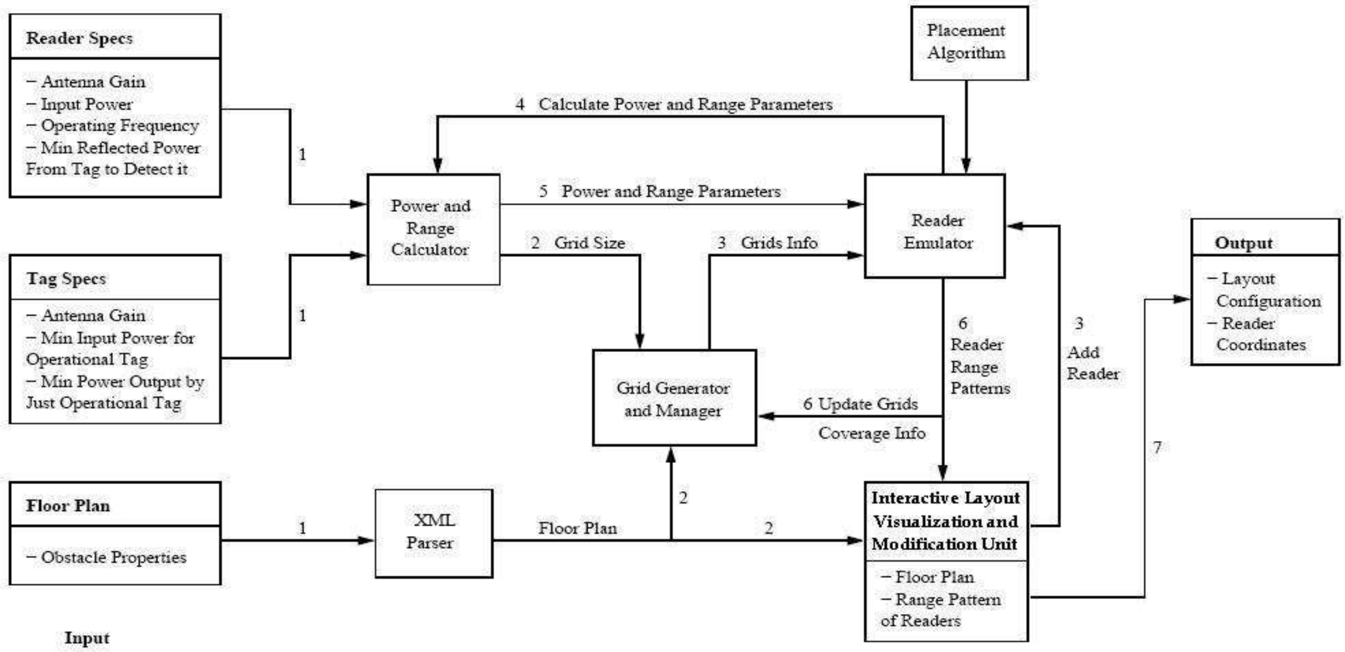

Fig. 1. *RFIDPlanner* Architecture

TABLE I
ATTENUATION VALUES

Type of Obstacle	Attenuation Value (dB)
Wooden partition	1
Brick wall	6

$$L = L_{FSL}(d) + \sum_{i=1}^{N} k_{wi} L_{wi} \quad (3)$$

where L_{FSL} (dB) is the free space loss for distance d(m) between transmitter and receiver antennas, k_{wi} is the number of walls of i-th type between transmitter and receiver antennas, L_{wi} is attenuation factor for i-th wall type and N is the number of wall types. Since we would be looking at a 2D model of the area, floor attenuations are not taken into consideration.

As RFID readers have small interrogation ranges, the effects of reflection, diffraction and scattering would not be substantial enough over the small area of coverage of a single reader. It would suffice to use representative measurements of propagation losses for calculating the range patterns. Therefore, empirical approach has been taken for propagation modeling in *RFIDPlanner* instead of a deterministic approach. We use a table similar to one shown in Table I for propagation loss calculations. More attenuation values for different type of materials can be found in [3] and [4].

E. Architecture

The architecture of *RFIDPlanner* is shown in Fig. 1. Some of the important aspects of the architecture are discussed below.

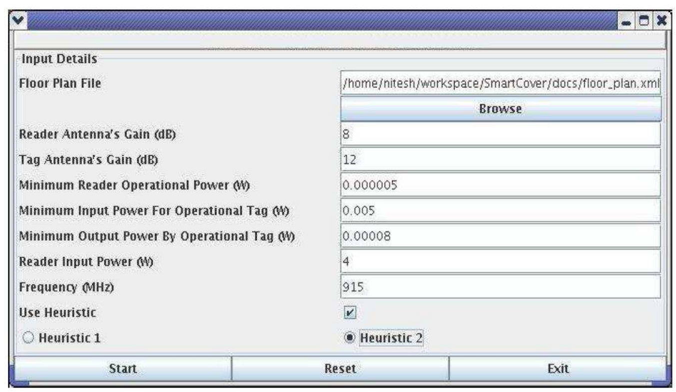

Fig. 2. Snapshot of the input GUI

1) Inputs: RFIDPlanner requires the floor plan of area to be covered and reader and tag specifications as inputs. A snapshot of the input GUI is shown in Fig. 2. The floor plan of the area to be covered is in the form of an XML file. It provides dimensions of the area to be covered and details of obstacles present in the given area. For every obstacle, the XML file contains its type, thickness and end coordinates of the line segment passing through the center of the obstacle along its length. This information, along with attenuation values from Table I, is subsequently used to determine attenuation at obstacles.

The required reader specifications are:

- Antenna gain
- Input power to reader
- Operating frequency of antenna
- Minimum reflected power input required at reader's an-

tenna for it to detect a tag

The required tag specifications are:

- Antenna gain
- Minimum input power required at its antenna, for the tag to be operational
- Minimum power output by a just operational tag

Reader and tag specifications, together, are required to calculate reader's range patterns.

2) Power and Range Calculator: This module forms a repository of formulae used for calculating power attenuation levels as well as the reader range. The input values for reader and tag specifications are stored in this module. Once these input values have been read, the module calculates the maximum reader range. This maximum reader range is used to determine the grid size. The reader emulator module queries this module while emulating a reader's range.

3) XML Parser: The XML parser is responsible to read the input floor plan XML file and supply the information to the gird generator module and display unit. Also, once the grids have been generated, the XML parser outputs an XML file containing the grid details.

4) Overlay Grid Structure: The *RFIDPlanner* Architecture is primarily based on an overlay grid structure. The area to be covered is divided into square grids of uniform size. The grid size is chosen such that the diagonal of the grid is equal to the maximum range of the readers to be deployed in the area.

Each grid contains details of sections of obstacles present in it. It also keeps track of readers covering it and the portion covered by that reader. While computing the reader range, we need to check for interference of radio waves with obstacles. Using the grid system ensures that we check for interference with only the obstacles present in the grids around, and including, the grid in which the reader is placed. Thus we avoid redundant checking of interference with far away obstacles. This makes the mathematical modeling of the effect of obstacles on reader range less computation intensive, and hence faster.

Given the floor plan and grid size, the *Grid Generator and Manager* generates the grid structure which is then used by the reader emulator while emulating reader ranges. It also provides the functionality of storing reader range information, generated by the reader emulator, in corresponding grids.

5) Reader Emulator: Reader Emulator sits at the heart of *RFIDPlanner* performing the most important task of emulating attenuation patterns. While emulating a reader's range we set two parameters for emulating the range. The first parameter is the *number of sections* in which the circular range of the reader can be divided into. Within each section, all points on the arc at a given distance from the center are assumed to receive the same level of power as the point on the center of that arc. In other words, it is assumed that all radio waves in a section will behave in the same manner as the radio wave passing through the center of that section. Thus, any obstacle that interferes with the radio wave passing through the center of a section, will be assumed to interfere with all

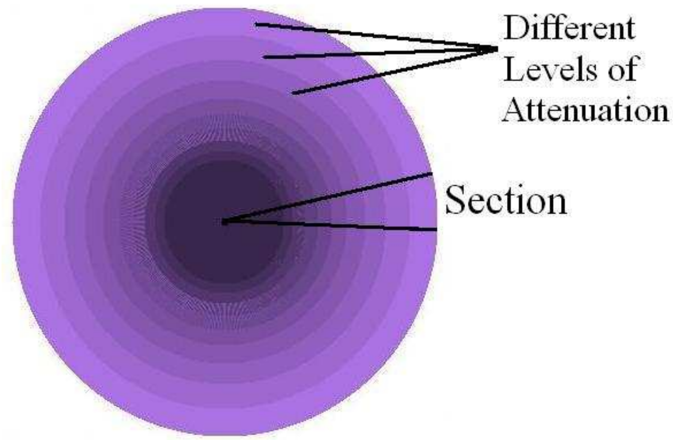

Fig. 3. Reader Range pattern in the absence of any obstacles

the radio waves in that section. This makes approximating the range attenuation patterns easier. The accuracy of this approximation depends on the *number of sections* in which the range is divided. It was observed that dividing the range into 360 sections delivered quite accurate results. Increasing the number of sections beyond this unnecessarily increased the time for computation and also gave rise to computational errors due to the extremely small size of sections.

Each section is further divided into a *number of attenuation levels*, which forms the second parameter. Each level has a minimum and a maximum allowed attenuation of power received from the reader. For the first level, minimum attenuation is 0, which will be at the reader itself. For an intermediate level, minimum attenuation allowed is the same as the maximum attenuation allowed for the previous level. For the last level, maximum attenuation allowed is the total free space attenuation attained over the range of the reader in absence of any obstacles. Each attenuation level is represented by a particular gradient of a color. As the level increases the color gradient assigned is lighter than the previous one. Thus, in the graphical visualization, gradients corresponding to the increasing attenuation levels give a fading effect to the reader's range, as shown in Fig. 3. This parameter is used to aid in visualizing range attenuation patterns. If an obstacle is present in the path of a radio wave, the attenuation in power of the radio wave just after the obstacle takes a sudden jump. This sudden increase in attenuation would be depicted clearly in the visualization. Dividing a section into 10 attenuation levels gives a fairly accurate approximation of range attenuation behaviour.

For each section we maintain a list of end coordinates corresponding to attenuation levels. The radio wave passing through the center of the given section passes through these set of coordinates. The start coordinate of a level is the same as the end coordinate of the previous level. For the first level, the start coodinate is the position of the reader itself.

While emulating reader range attenuation behaviour for a

```
// maximum atttenuation of power allowed
maxAtten = GetMaxAttenuation();
attenInterval = maxAtten / noOfAttenLevels;
for(i=0; i< noOfSections; i++) {
        // total attenuation becasue of obstacles in this section. keeps adding up
        obsAtten = 0;
        start = GetReaderPosition();
        for(j=0; j<noOfAttenLevels; j++) {
                maxAttenForLevel = (j+1)*attenInterval − obsAtten;
                freeSpaceDist = GetFreeSpaceDist(maxAttenForLevel);
                end = FindCoordinate(freeSpaceDist, i);
                while(maxAttenForLevel > 0) {
                        obstacle = FindNextInterferingObstacle(i, start, end);
                        if(obstacle == null) {
                                break;
                        }
                        else {
                                dist = FindDistance(reader, obstacle);
                                freeSpaceAtten = GetFreeSpaceAtten(dist);
                                if(freeSpaceAtten + obstacle.getAtten() >= maxAttenForLevel) {
                                        end = FindInterferenceCoord(i, obstacle);
                                        break;
                                }
                                else {
                                        start = FindInterferenceCoord(i, obstacle);
                                        maxAttenForLevel −= obstacle.getAtten();
                                        freeSpaceDist = GetFreeSpaceDist(maxAttenForLevel);
                                        end = FindCoordinate(freeSpaceDist, i);
                                }
                                obsAtten += obstacle.getAtten();
                        }
                }
                AddLevel(i,j,end);
                start = end;
        }
}
```

Fig. 4. Algorithm for Reader range emulation

particular attenuation level in a given section, we basically calculate the start and end coordinates of this attenuation level. As mentioned earlier, the start coordinate is the same as the end coordinate for previous level, hence there is no need to find it again. Every attenuation level has a maximum allowed attenuation of power received from the reader at its end coordinate. This end coordinate would lie at the maximum distance from the reader in absence of any obstacle. If any obstacles are present in this or any previous attenuation level of the same section, the attenuation due to these obstacles will also have to be taken into consideration. Thus, the total free space attenuation would be reduced and the end coordinate would be shifted towards the reader. This would therefore result in reducing the effective range of the reader. A more detailed version of this algorithm for calculating a reader's range pattern is shown in Fig. 4.

A placement algorithm can be used to automatically generate appropriate reader positions, using the reader emulator, for completely covering the given area. Two such algorithms have been described in Section III.

6) Interactive Layout Visualization and Modification Unit: The interactive layout visualization and modification unit forms another important part of *RFIDPlanner*. The visualization unit provides an interactive GUI which displays reader range patterns and the floor plan of the given scenario, showing distinct obstacles in different colors for easy visualization as shown in Fig. 5. A zoom feature has been implemented to enhance visualization of reader range patterns.

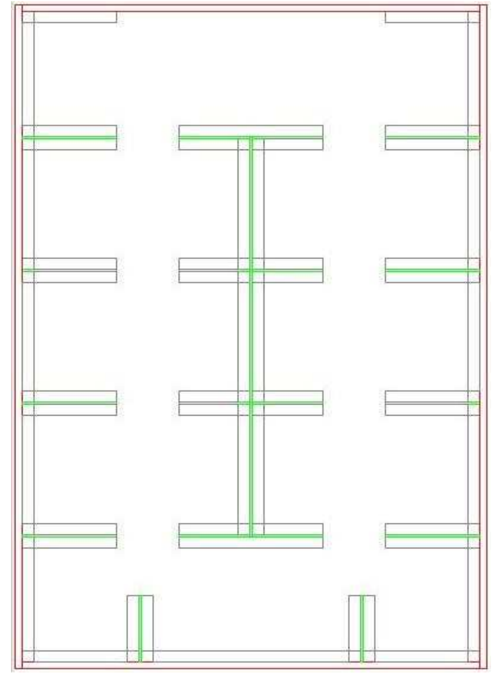

Fig. 5. Floor plan visualization

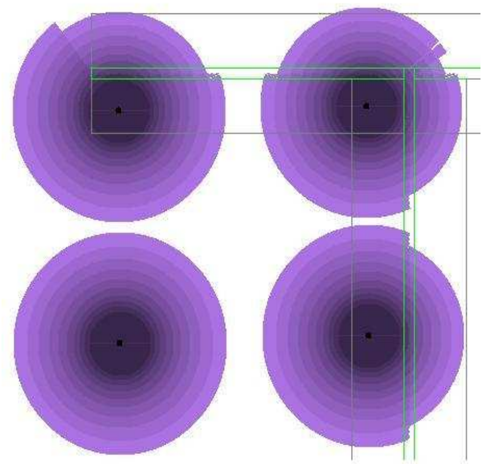

Fig. 6. Range attenuation patterns

The interactive modification unit is tied up with the reader emulator to perform the main tasks of adding new readers and moving or deleting existing ones. Users can add new readers at any location in the given area, simply by clicking on the desired location. This information is then passed on to the reader emulator. Once range attenuation patterns are calculated by the reader emulator, the reader and its range attenuation patterns are displayed as shown in Fig. 6. Users can now move this reader to other locations to observe any differences in the range attenuation patterns at the new and old reader positions. The entire process has been made interactive, allowing users to modify the reader layout during runtime.

7) Outputs: The final output of *RFIDPlanner* is a layout configuration of RFID readers, including reader coordinates, in the given area.

8) Architecture Flexibility: The *RFIDPlanner* architecture has been designed and developed in a flexible manner to allow changing configuration of different modules as well as addition of independent modules to enhance the functionality of the tool. For example, it is possible to build in support for different type of readers and tags available in the market. The database of different types of obstacle materials and their corresponding empirically determined propagation loss data can be easily expanded to include new information. The architecture also allows addition of independent heuristics which can generate sample configurations of reader placement according to given scenarios. These configurations can then be visualized using *RFIDPlanner* and appropriately modified to suit requirements. The architecture thus allows *RFIDPlanner* to be easily adaptable to different user needs. In the following section we present a case study of a shopping mall scenario and two different heuristics, designed keeping in mind the requirements of a shopping mall, which have been implemented using *RFIDPlanner*.

III. Shopping Mall Scenario : A Case Study

A Shopping mall usually consists of a large hall, which has several wooden or plastic partitions for different items and brands. Each partition, in turn has several steel or glass shelves on which the items are displayed. In such a scenario, tagged items are confined to the shelves, and hence we need to ensure that all the shelves present in the given area are fully covered by the RFID reader network.

Assuming a similar given scenario, which has a large hall made of brick walls and consisting of several wooden partitions and steel shelves, we have designed two different heuristics to automatically generate reader placement configurations ensuring complete coverage of the area where tagged items will be placed $i.e.$ shelves. Walls, partitions and shelves form the set of obstacles (with different propagation loss properties) that we will be dealing with. Thus all obstacles of type shelf will need to be fully covered. These heuristics are described in detail in the following sections.

A. Heuristic H1

Heuristic H1 has been designed with the aim of validating the basic functioning of *RFIDPlanner* and of using it for comparison with other heuristics. It depends largely on different modules of *RFIDPlanner* such as the overlay grid structure and reader emulator to come up with placements of readers ensuring complete coverage of the desired area. The algorithm for H1 is shown in Fig. 7.

```
gridsToBeCovered = FindGridsContainingShelves();
while(gridsToBeCovered != null) {
    firstUncoveredGrid = FindFirstUncoveredGridContainingShelfEdge();
    reader = placeReader(firstUncoveredGrid);
    partiallyCoveredGrids = FindPartiallyCoveredGridsContainingShelves(reader);
    while(partiallyCoveredGrids != null) {
        newReader = placeReader(partiallyCoveredGrids.get(0));
        newPartiallyCoveredGrids = FindPartiallyCoveredGridsContainingShelves(newReader);
        partiallyCoverdGrids.add(newPartiallyCoveredGrids);
        girdsToBeCovered.remove(partiallyCoveredGrids.get(0));
        partiallyCoveredGrids.remove(0);
    }
}
```

Fig. 7. Algorithm for Heuristic H1

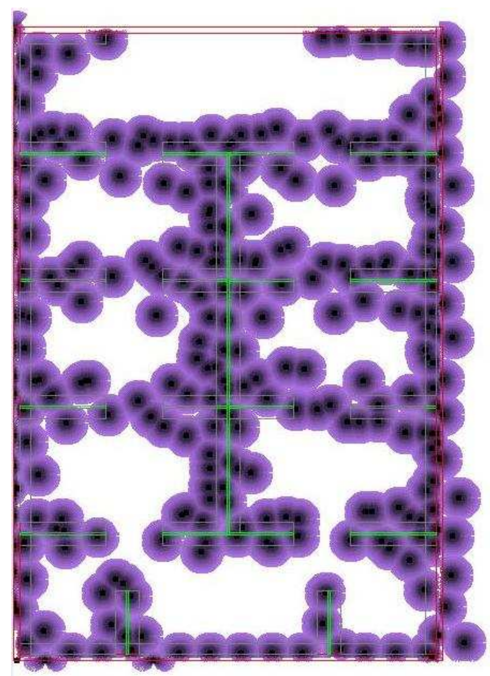

Fig. 8. Readers placement configuration using Heuristic H1

H1 starts by selecting the first grid that contains an edge (along the breadth) of a shelf and placing a reader on the middle of this edge. The *RFIDPlanner* reader emulator is then used to emulate the range pattern and determine which all grids are partially or fully covered by this reader. The partially covered grids are added to a global FIFO list. These partially covered grids are then analyzed one by one. If a partially covered grid does not contain any shelf, no readers are added to further cover the grid. However, if there are shelves in this grid, another reader is placed so as to cover the entire grid while trying to minimize interference with readers already covering the grid. The new grids which are now partially covered by the new reader would be added to the FIFO list and the procedure is repeated until all the shelves are covered completely.

A sample configuration of readers placement in the floor plan shown in Fig. 5 generated using H1 is showed in Fig. 8.

Apart from validating the basic functioning of *RFIDPlanner*, this heuristic also demonstrates the ease with which underlying modules of *RFIDPlanner* can be accessed and utilized by heuristics which might be added later to *RFIDPlanner*.

```
shelves = GetListOfShelves();
for(i=0; i<shelves.size(); i++) {
    shelf = shelves.get(i);
    if( !isCovered(shelf)) {
        adjacentShelf = FindAdjacentShelf(shleves.get(i));
        obstacle = ObstacleBetween(shelf, adjacentShelf);
        if((adjacentShelf== null) || isCovered(adjacentShelf)) {
            placeReadersAlongAxis(shelf);
        }
        else if(attenuatedReaderRange(shelf, adjacentShelf) <
                        adjacentShelf.thickness + obstacle.thickness()) {
            placeReadersAlongAxis(shelf);
        }
        else {
            placeReadersAlternatelyAlongObstacle(shelf,adjacentShelf,Obstacle);
        }
    }
}
```

Fig. 9. Algorithm for Heuristic H2

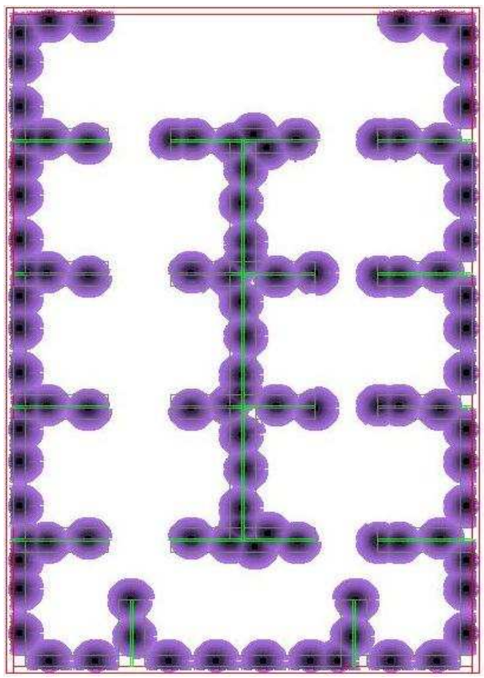

Fig. 10. Readers placement configuration using Heuristic H2

B. Heuristic H2

The design of heuristic H2 is an attempt to capture the logic we would normally use while placing RFID readers manually in the given scenario. For the sake of algorithmic convenience we modify the floor plan such that if any two shleves are touching each other, we insert an imaginary obstacle with zero thickness and zero propagation loss, between the two.

The first objective of this heuristic is to ensure that there is no need for creation of additional infrastructure for deploying readers. In other words, it should be made sure that the readers are placed on the existing obstacles. Secondly, if there are adjacent shelves, we try to place a RFID reader such that it covers appropriate portions of both the shelves, thus minimizing the number of RFID readers required. The algorithm for H2 is shown in Fig. 9.

To achieve these objectives, instead of considering all the grids containing shelves as was done in H1, we directly make a list of all the shelves. These shelves are now analyzed one by one. If a shelf is not covered, we look for adjacent shelves to the given shelf. If uncovered adjacent shelves are not found, we simply place RFID readers along the axis of the shelf (the line running through the center of the shelf along its length) ensuring minimum interference with earlier placed reader, while covering the entire shelf.

In case an uncovered adjacent shelf is found, we first check for the effect of propagation loss due to the obstacle between the given shelf and the adjacent shelf. In this case a RFID reader is placed next to the obstacle, on the given shelf. If the effective interrogation range is less than the combined thickness of the obstacle and the adjacent shelf, then it is not possible for the RFID reader to cover parallel portions of the shelf and the adjacent shelf simultaneously. Thus we place the RFID reader along the axis of the self and ignore its coverage effect on the adjacent shelf.

If the interrogation range is greater than the combined thickness of the obstacle and the adjacent shelf, we place the RFID reader next to this obstacle on the given shelf. The next RFID reader is placed again next to the same obstacle but this time on the adjacent shelf, such that the interference is minimum between the ranges of the two RFID readers. Readers are thus placed in an alternate manner on the shelf and the adjacent shelf, aligned to the obstacle between the two, till atleast one of them is fully covered.

This procedure is repeated for all the shelves, thus ensuring that all the desired area is fully covered. A sample readers placement configuration for the floor plan shown in Fig. 5 generated using H2 is shown in Fig. 10. H2 makes use of the overlay grid structure of *RFIDPlanner* to determine the adjacent obstacles and adjacent shelves. It also uses the module used for propagation loss and range calculations.

C. Results and Comparisons

It has been shown that coming up with an optimum configuration for coverage planning is an NP hard problem [5]. Thus, a good way to judge the performance of a heuristic implemented in *RFIDPlanner* would be to compare the number of RFID readers placed by the heuristic to cover the entire given area with the number of RFID readers that would be placed manually. We assume that while placing RFID readers, the user would use proper discretion to avoid redundancy and interference. The results for deployment of RFID readers in the area shown in Fig. 5 are shown in Table II.

With manual placement we were able to cover the given area with 85 RFID readers. H1, which is a naive heuristic developed mainly for the purpose of validation, came up with a configuration containing 317 RFID readers. On the other hand, H2, which follows a more practical approach, delivered better results. The configuration arrived at using H2 deployed 102 RFID readers to cover the same area. The configuration is quite similar to the configuration got by manual placement of RFID readers.

TABLE II

DEPLOYMENT RESULTS FOR AREA SHOWN IN FIG. 5

Method of Deployment	No. of RFID Readers	Ratio of No. of Readers Deployed Using Current Method to No. of Readers Deployed Manually
Manual Placement	85	1
Heuristic 1	317	3.73
Heuristic 2	102	1.2

For the algorithm for H1, in the worst case scenario, all the shelf containing uncovered grids will be scattered away from each other, $i.e.$ when a reader is placed to cover one of these grids, the partially covered grids thus generated will not contain any shleves. In this case the function $FindFirstUncoveredGridContainingShelfEdge()$ will be called for each grid containing a shelf. This function has a complexity of $O(n)$, where n is the total number of grids. The complexity for placing a reader is $O(1)$. A reader can at a time cover a maximum of 14 grids. Therefore, the complexity for finding partially covered grids containing shelves is $O(14)$. Thus, the total complexity in this case will be $O(n(n+1+14))$, or $O(n^2)$.

In the best case scenario, all the grids containing shelves will be nearby. Here, when a reader is placed in one of the grids, some or all of the resulting partially covered grids will contain shelves. Readers will be placed to cover these grids, which would intern generate more such shelf containing partially covered grids and so on untill all the required grids are covered. In this case, the function $FindFirstUncoveredGridContainingShelfEdge()$, with complexity $O(n)$ is called only once. For covering each partially covered grid, complexity for placing a reader is $O(1)$ and for finding partially covered grids containing shelves is $O(14)$. Therefore, complexity for covering all partially covered grids is $O(n(1+14))$, $i.e.O(n)$. Thus, complexity for the complete algorithm is $O(n+n)$, $i.e.O(n)$.

In the algorithm for H2, each shelf is analyzed once. The complexity for finding the adjacent shelf is $O(m)$, where m is the total number of shelves in the given area. Once an adjacent shelf has been found, the complexity for finding the obstacle between the two shelves is $O(1)$. The complexity for placing readers, either along the axis of the shelf or along the obstacle, is $O(l)$ where l is the length of the shelf. Thus, the overall complexity becomes $O(m(m+l))$. If we consider the length of shleves to be constant, the complexity becomes $O(m^2)$.

IV. CONCLUSION

In this paper we explored the need for RFID domain specific coverage planning tools. We described in detail the design and architecture of *RFIDPlanner* - a coverage planning tool for RFID networks, including the algorithm used for emulating RFID reader range patterns. Further, the flexibility of its architecture was demonstrated by utlizing different modules of *RFIDPlanner* to develop two different heuristics for coming up with RFID readers placement configurations. The algorithms for these heuristics were explained in detail and result of their implementations were compared.

The *RFIDPlanner* architecture provides a basic framework for a RFID coverage planning tool. It presents a lot of flexibility for further addition of features, such as support for slant obstacles, new heuristics for different scenarios and direct support for different RFID readers and tags available in the market eliminating the need for the end user to be familiar with RFID reader and tag specifications. With such functionality enhancements, the scope for application of *RFIDPlanner* can be increased and thus utilized for coverage planning in much complex scenarios.

ACKNOWLEDGMENT

The authors would like to thank Persistent Systems Private Limited for funding this project.

REFERENCES

[1] K. Finkenzeller. “RFID Handbook - Fundamentals and Applications in Contactless Smart Cards and Identifi cation”, *Chichester: John Wiley, Leipzig, dritte edition*, 2003.

[2] S. Zvanovec, P. Pechac and M. Klepal, “Wireless LAN Networks Design: Site Survey or Propagation Modeling?”, *Radioengineering, Vol. 12, No.4*, December 2003.

[3] T. Rappaport, “Wireless Communications: Principles and Practice”, *Prentice Hall*, 1996

[4] B. Silberberg, “Technical State of 868-870 MHz Radio Modules in the SRD Band”, *AUBE’01, 12th International Conference On Automatic Fire Detection*, March 25 - 28, 2001. http://fi re.nist.gov/bfrlpubs/fi re01/PDF/f01063.pdf

[5] S. Anusha and S. Iyer, ”RFIDcover: A coverage planning tool for RFID networks with mobile readers”, *1st International Workshop on RFID and Ubiquitous Sensor Networks (USN)*, Nagasaki, Japan, Dec 2005.

[6] S.J. Fortune, D.H. Gay, B.W. Kernighan, O. Landron, R.A Valenzuela, and M. H. Wright, “WiSE Design of Indoor Wireless Systems: Practical Computation and Optimization”, *IEEE Comput. Sci. Eng., vol. 2, pp. 58–68*, March 1995.

[7] SpectraGuard Planner, *AirTight Networks.* http://www.airtightnetworks.net/productsandservices/sg_planner.html