Patient Service System

CS308 Project Presentation I

Group Members :

Pritish Kamath	(08005004)	Rohit Saraf	(08005040)
Ashish Mathew	(08d10035)	Vivek Madan	(08005035)

under the guidance of
Prof. Kavi Arya

Patient Service System - Problem Statement

1. Automatic attendants which would serve patients' requests such as supplying water or patient specific medicines etc.
2. These attendants should be summonable using an easy to use interface by the patients.
3. The patients should be served in a fair-share manner and the requests should satisfy bounded wait.

Patient Service System - Description

1. Each patient has a TV remote with which he/she can send a request to the server.
2. On receiving a request, the server sends a bot to serve the patients' requests. This is communicated to the patient via an SMS.
3. The server ensures reliable communication (via Zigbee) with all the serving bots and the patient controller.
4. The server design must be scalable to serve a large number of patients, and must be free of deadlocks, race conditions and must have maximum concurrency.
5. If a bot is blocked on it's path to the patient, then the server informs the security about the block, via an SMS.

Functional Requirements

1. Central Server
   - Patient requests should be communicated to the centralized server through a wireless medium.
   - The centralized server would send attendants to serve the patients' requests or communicate to doctors about the urgency of the situation (through SMS or otherwise), whichever applicable.
   - It should run a scheduling algorithm to manage the robotic attendants so as to serve the patients in a least possible time.
2. Robotic Attendants
   - The robot should follow a fixed path indicated by white lines.
   - It should stop in presence of an obstruction and alarm the guards about its prolonged obstructed movement and send an SMS (through the server) to inform the security.
   - It should be able to serve multiple patients, on request.

Work Execution (1/2)

- Got Zigbee communication working with Java. (2 days)
- Making Arena and perfecting White Line Follower (7-fold algorithm with error recovery) (2 days)
- Designing Polling based communication protocol : both server and bot. (1 day)
- Creating MySQL Database for patient requests, and designing Bot Locomotion algorithm (ensuring no deadlocks and bot-bot collisions). (2 days)
- Simulation (with rigorous testing - independent of the bot) of the server. Checking for deadlocks and race conditions. (2 days)

Work Execution (2/2)

- Taking input from *TV remote* for patient requests. Designing a *Protocol independent Learning based Algorithm* for capturing inputs. (2 days)
- Sending *SMS* through python script. (1 day)
- Writing scripts for configuring the system for setting up all necessary softwares. (1 day)
- Documenting all code (using Doxygen tool), writing readme file and making the final presentation (1 day).

Work Division (1/2)

Work/Critical Tasks	**People responsible**
Zigbee communication with Java	Vivek Madan, Rohit Saraf, Ashish Mathew
Making Arena Perfecting Line Follower	Pritish Kamath, Rohit Saraf
Implementing Server Side Polling based Communication Protocol	Vivek Madan, Ashish Mathew
Implementing Bot Side Polling based Communication Protocol	Pritish Kamath, Rohit Saraf

Work Division (2/2)

Work/Critical Tasks	**People responsible**
MySQL Database for patients Bot Locomotion Algorithm (server) Simulation	Vivek Madan, Ashish Mathew
Taking input from TV remote Sending SMS via Python Script (internet)	Pritish Kamath, Rohit Saraf,
Configure Scripts and Makefile	Rohit Saraf
Documentation	All

Project Completion Date : 5th April 2011

Bot FSM

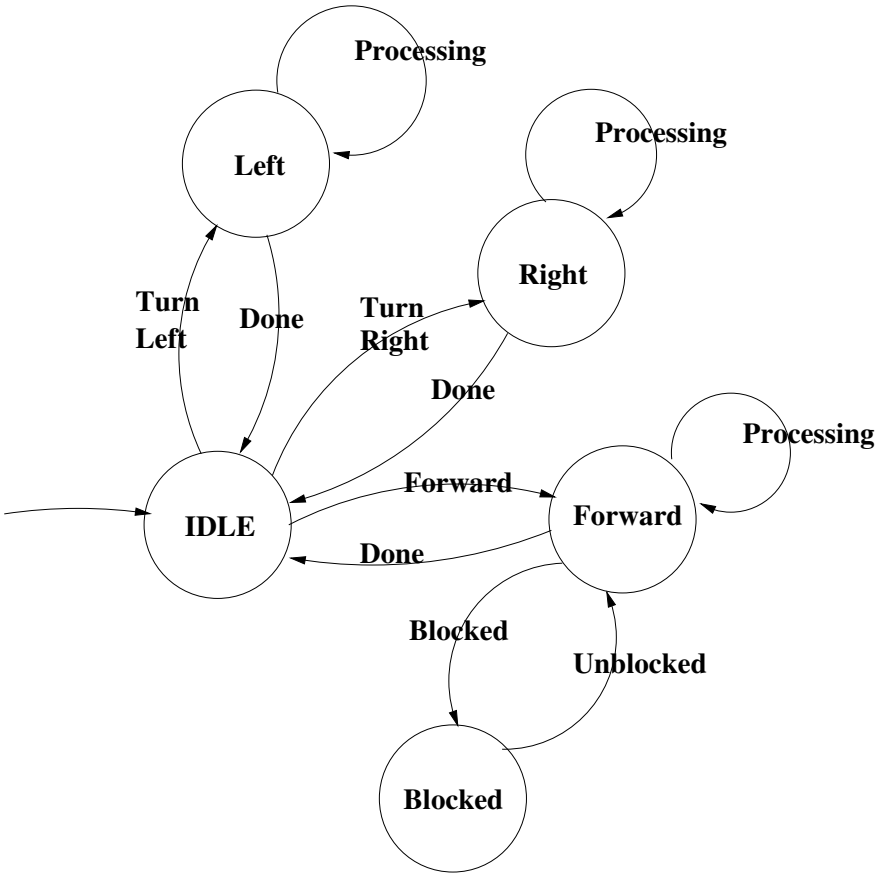

Figure: Service Bot FSM

Innovation and Challenges (1/2)

- Server runs in Ubuntu. Bot AVR-programming done in Ubuntu.
- Zigbee communication using *Java*.
- MySQL database. Makes the project scalable for large number of patients.
- Correction and Recovery while following the white line. Smooth running of the bot, (without any extra squares added at intersections).
- Polling based communication implemented to avoid interference between serving bot and patient bot.

Innovation and Challenges (2/2)

- Maximum parallelism and concurrency to avoid time/resource wastage. (Multithreading which is not concurrent like pthread, but actually parallel).
- Testing everything and making the design modular so that the different parts have least to do with one another. Done using virtual simulation of bots.
- Ensuring no deadlock between multiple bots.
- Getting TV remote to work using a Protocol independent learning based algorithm.
- Sending SMS to phone.
- Efficient Scheduling Algorithm.

Problems Faced and Solutions (1/2)

Problems faced	**Devised Solutions**
Line following not working reliably	7-fold algorithm employing Error Correction and Recovery
Simultaneous Zigbee communication with two other Zigbee modules	Polling based protocol
Delay required after sending data via Zigbee causes problem in execution of Graph algorithm	Multithreaded server : Separate thread for each send operation

Problems Faced and Solutions (2/2)

Problems faced	**Devised Solutions**
TV remote protocol not reliable	Use Protocol independent Learning based algorithm
IR-interrupt conflicting with USART sensor	Do IR checking in a busy wait manner.
A large files causing problems in compilation and setup	Configure Scripts and Makefile

Arena

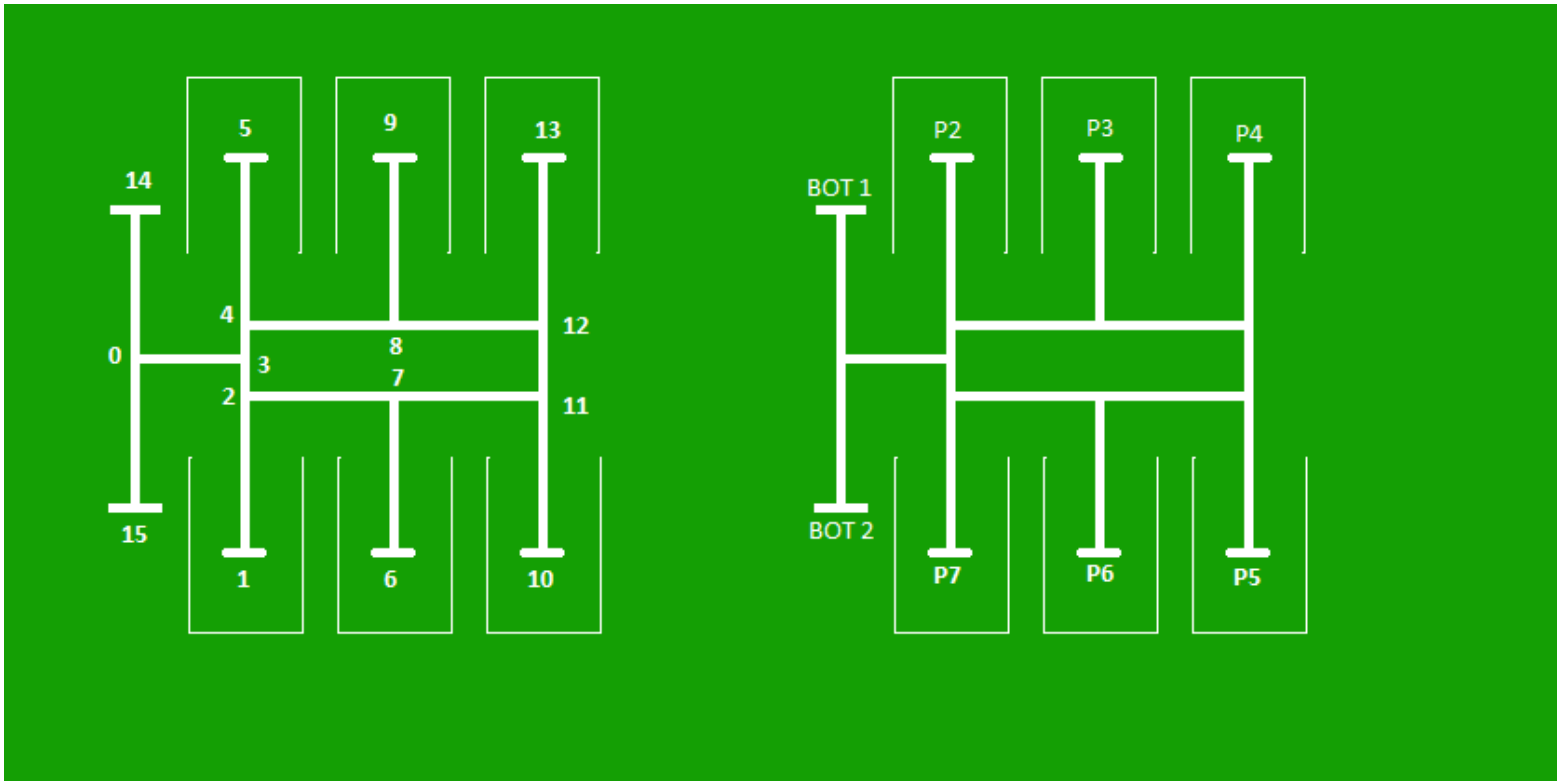

Figure: Arena

Modular Testing

- Testing multiple Zigbee client - single Zigbee server protocol.
- Virtual bot simulation and testing of server.
- Testing for deadlocks and race conditions.
- Server independent testing of serving bots.
- Independent TV remote testing.

Combined Testing

- Correct patient-request is served by the attendant.
- Attendant stops in presence of obstruction and beeps in case of prolonged obstruction.
- 6 patient controllers using TV remote.
- Simultaneous requests allowed (served in fair-share FIFO order).

Code Reusability

1. Server code : Completely object oriented (Java).
2. Modular Design : Modules independent of each other, hence can be changed without changing others.
   - A different arena can be stored by changing *Graph.java*.
   - Polling protocol can be changed by changing *PollingThread.java*.
   - *Properties* file stores all user dependent constants that can be changed without going through/compiling the code.
3. Server is bot-hardware independent. Relies on only a serial communication protocol (not necessarily Zigbee) with the bot.

Future Directions

- Can be adapted to suit hospitals like Tata Memorial Hospital.
- Attendants can be suited to check patient condition (such as glucose bottle levels) and take appropriate action (refill the glucose bottle).
- Can be used to help in house-hold activities (such as watering plants, etc.). TV remote based interface makes it convenient to use.