An Automatic Management of Car Parking using Smart LAN and Relational Database Technologies

Subhi H. Hamdoon, PhD
Sur College of Applied Sciences, Ministry of Higher Education, Sultanate of Oman.

Elsayed Soleit
Sur College of Applied Sciences, Ministry of Higher Education, Sultanate of Oman.

ABSTRACT
Smart network technologies have received great attentions and are widely applied in Control and Management. The joint implementation of these types of Networks and Database administrations adds intelligible depth and self-learning capabilities. The Smart Network is based on multipurpose distributed sensors which are implemented in allocated spatial locations. The role of these sensors is to create parametric and visual data related to the physical locations of these sensors. The captured data is formatted as data frames. Each data frame includes different fields (Sensor ID, Spatial location, Control bits (enable/disable), Time, Image data, etc.). Each sensor is connected to wireless LANs that are connected directly to central Data Server. The captured data frames are processed, analyzed and classified into data groups. The proposed model is implemented and tested in the simulation phase. The implemented system is operable and the output results are valuable and encourage the real time implantation of the simulated model.

General Terms
Networks, Database administration, Sensors, Data frames, Wireless LAN, Data server, Network topologies et. al.

Keywords
Smart Network Technologies, Multipurpose distributed sensors, Spatial locations, Parametric data, Smart distributed sensors, Intelligent LAN, Smart Client/Server LAN, Distributed wireless smart sensors, Central wireless receiver, Frame Check Sequences.

1. INTRODUCTION
The intelligent computer networks technologies are of great interest and are applied in several fields. The intelligent computer networks topologies are based on the joint operation of the smart distributed sensors and the usual wire or wireless LAN topologies under the management administration of central data server [1-3].

In this paper a proposed joint scheme of intelligent LAN is introduced as depicted in Fig.1. The scheme structure is based on the smart distribution sensors and wireless client/server LAN topology.

Relational database is used to design this system and a (PL/SQL Oracle 10g) is used to implement the process of the proposed model [4-5].

The proposed main procedures include:
- Add (insert) new smart sensor.
- Remove (delete) existing smart sensor.
- Modify (alter) the characteristics of specific smart sensor, group of smart sensors or even all sensors in the network.
- Release (enable) smart sensor.
- Mark smart sensor as busy (disable).
- Accounting (billing).
- Recovery procedures (Backup/Restore).
- Advertisements.

This paper includes four sections. Section two presents the proposed joint model of an intelligent computer client/server LAN. Section three explains the system implementation. Section four highlights the main conclusions.

2. A PROPOSED JOINT INTELLIGENT LAN MODEL
A proposed Joint model of a Smart Client/Server LAN is exhibited in Fig.1. A relational data base is introduced to automatically manage Car Parking activities.

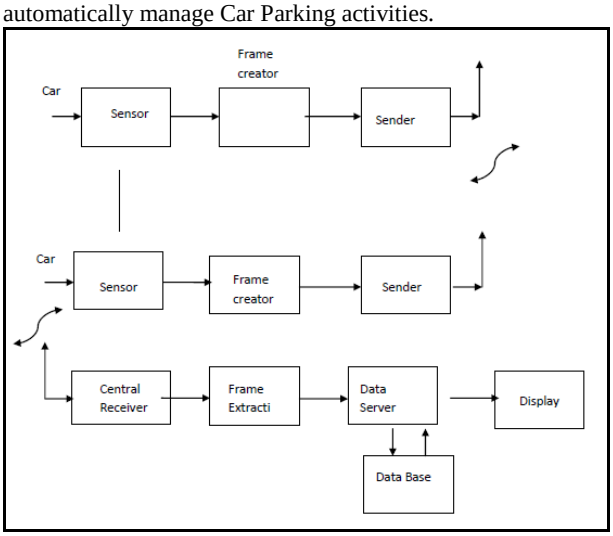

Fig 1: Joint Model of Smart LAN

The proposed Joint model of a Smart LAN comprises of the following modules:
- Distributed wireless smart sensors.
- Central wireless receiver.
- Data server.

2.1 Distributed wireless smart sensors
The wireless smart sensors are responsible to generate and send data frames that include the sensor ID, the car data and trigger pattern to the data server via the central receiver at the time of car occupation of the sensor location. Each sensor location (x-y coordinates) is assigned a priory in the data base. The frame format is shown in Fig.2.

Fig 2: Sensor Data Frame

The different frame formats depicted in Fig.2 include the following fields [4-6]:

Start: 12 bits pattern that refers to the starting of frame. This pattern is used also for frame synchronization between the transmitted and received frames to decrease the bit error rate.

Source ID: The length of this field varies according to the type of sensor. This field identifies the sensor number and the position coordinates with respect to the reference point.

Control: This field indicates the availability state of the car parking locations. The car parking locations state may be Empty or Occupied (enable/disable).

Time: This field includes two subfields (the arrival initial time and the departure time) for any specific car.

Data: The data field includes the collected data from the Smart car plate. This data comprises the car number and car characters. Any other information about car can be obtained from these two fields (composite primary key).

FCS: This field contains the frame check sequences which enables the error detection of the transmitted frames.

END: This field contains a known pattern that determines the end of frame.

2.2 Data Server

The function of the data server is to manage and administer all the functions of intelligent computer networks. It runs out the required software programs and Database procedures. The software programs are network operating systems, the required protocols and the data base procedures. The data base model will be explained in the next section[6].

3. IMPLEMENTATION OF THE PROPOSED MODEL

3.1 Database Tables

In this system the following four Database tables are used at the implementation phase:

1. PARKING_COST table

This table includes the attributes necessary about the total cost, start and departure time car identification (car characters and car_ number) and also the sensor identifier.

2. Sensor table

It includes the following attributes: sensor status either enables or disable, car identification attributes (car_characters and car _number) and also the current sensor identifier.

3. SENSOR_DETAIL table

The location (X/Y coordinates) and general information about each sensor are stored in this table; the sensor identifier is used as a primary key and can be referenced by other tables.

4. 4-TRIGGER_TABLE table

The data sent by camera as snapshot are stored in this table, to be used by trigger procedures, the camera sends two snapshots. The first one is sent immediately after entering the car to the park and the second one is sent immediately after the car departure. After each snapshot a trigger (insert or update trigger) becomes active and the required procedures run automatically to store the necessary information about the assigned car (car_characters and car_number) in addition to the start/departure time. This data will be used in the calculation procedure to fix the parking cost for the identified car according to current prices.

3.2 Database Package

The system includes many PL/SQL procedures, function and triggers. One package is created to deal as single unit. A brief description of the implemented package is as follows:

1. **SNAP_SHOT procedure:** this is used as a simulation of camera snapshot. Data sent by camera is stored in this table.
2. **IN_SENSOR procedure:** receives snapshot data (char_characters and car_number) together with start time and store it in SENSOR table, disable the current sensor.
3. **SEN_UPDATE procedure:** is used as intermediate link for trigger process. when car leave the park, the camera immediately sends the second snapshot and the sent data are used to update the departure time in the trigger table (initially departure time was set to null value).
4. **TIME_INTERVAL procedure:** checks start and departure time for each car and given sensor, find the elapsed time.
5. **OUT_SENSOR procedure:** the DEPART_TIME will be stored into sensor, and trigger PARKING_COST_CALC procedure.
6. **PARKING_COST_CALC procedure:** cost of parking for each car, according to elapsed time is calculated by this procedure.
7. **PARKING_RESET procedure:** used to initialize current sensor (null all current data) for the identified sensor except sensor identifier.
8. **Time_Ratio function:** the cost is determined according to car elapsed time, this is a dynamic procedure and calculates the parking cost according to given current prices (can be saved in a database table).
9. **PARKING_INIT trigger:** when any camera sends input (first snapshot), the trigger will be become active and trigger automatically the In_sensor procedure.
10. **PARKING_FINISH trigger:** automatically trigger both out_sensor procedure and PARKING_COST_CALC procedure. It becomes active immediately after camera sends output (second snapshot).

The performance of the proposed algorithm is measured and evaluated in the simulation design phase. The followings are considered as performance measures and evaluations:
1.System Complexity; 2.Accessibility; 3.Transaction Time; 4.Scalability; 5.Fault Tolerance

The source code is given in Appendix.

4. CONCLUSIONS

The intelligent computer networks are based on observable and controllable LAN topologies. These types of networks

offer several intricate interactive services. In this paper, the intelligent network is applied to manage and administrate a car park. The proposed system is implemented in the simulation phase. The system performance encourages implementing the proposed model in real time.

5. REFERENCES

[1] Kang Porlin and eta, "Smart messages: A Distributed Computing platform for networks of embedded system", The computer journal, vol. 47, No. 4, 2004.

[2] Kevin Loney Oracle Database 10g: The Complete Reference, Oracle Press, 2004.

[3] Stalling William, "Computer networking with internet protocols and technology", Pearson Prentice Hall Education, Inc., 2004.

[4] M. Kifer, A. Bernstein and P. M. Lewis. Database Systems: An Application Oriented Approach, second edition. Addison-Wesley, 2005.

[5] Stalling William "Wireless Communication & networks", 2nd edition. Pearson Prentice Hall, 2005.

[6] Tomsho Grey, Tillel Ed and Jonson David "Guide to networking essentials", 5th edition, Thompson Course Technology, 2007.

[7] Belbachir Ahmed, "Smart Cameras", Springer; 1 edition, December 2, 2009 ISBN: 978-1-4419-0952-7.

[8] Charmette Baptiste and eta, " Efficient planar features matching for robot localization using GPU", Fourth ACM/IEEE International Conference on distributed Smart Cameras, August 31, 2010 - September 4, 2010, Computer vision central, Quirical LLC.

Appendix-A

```
-------------------------------------------------------
-- Source Code
-- An Automatic Management of Car Parking Using Smart LAN and Relational Database Technologies
-------------------------------------------------------
--  DDL for Table PARKING_COST
-------------------------------------------------------
  CREATE TABLE "PARKING_COST"
   (    "VEHICLE_CHARACTERS" CHAR(2),
        "VEHICLE_NUMBER" NUMBER(6,0),
        "SEN_ID" NUMBER(5,0),
        "TOTAL_COST" NUMBER(6,2),
        "START_TIME" DATE,
        "DEPART_TIME" DATE
   ) ;
-------------------------------------------------------
--  DDL for Table SENSOR
-------------------------------------------------------
  CREATE TABLE "SENSOR"
   (    "VEHICLE_CHARACTERS" CHAR(2),
        "VEHICLE_NUMBER" NUMBER(6,0),
        "START_TIME" DATE,
        "DEPART_TIME" DATE,
        "SEN_STATUS" CHAR(1),
        "SEN_ID" NUMBER(5,0)
   ) ;
-------------------------------------------------------
--  DDL for Table SENSOR_DETAIL
-------------------------------------------------------
  CREATE TABLE "SENSOR_DETAIL"
   (    "SEN_ID" NUMBER(5,0),
        "X_COR" CHAR(2),
        "Y_COR" CHAR(2),
        "PARK_INFO" CHAR(100)
   ) ;
-------------------------------------------------------
--  DDL for Table TRIGGER_TABLE
-------------------------------------------------------
  CREATE TABLE "TRIGGER_TABLE"
   (    "SEN_ID" NUMBER(5,0),
        "VEHICLE_CHARACTERS" CHAR(2),
        "VEHICLE_NUMBER" NUMBER(6,0)
   ) ;
-------------------------------------------------
REM INSERTING into TRIGGER_TABLE  Test Data
Insert into TRIGGER_TABLE (SEN_ID,VEHICLE_CHARACTERS,VEHICLE_NUMBER) values (1,'a ',1122);
-------------------------------------------------
--   END DATA FOR TABLE TRIGGER_TABLE
-------------------------------------------------
-------------------------------------------------
--   DATA FOR TABLE SENSOR
--   FILTER = none used
-------------------------------------------------
REM INSERTING into SENSOR  Test Data
Insert into SENSOR (VEHICLE_CHARACTERS,VEHICLE_NUMBER,START_TIME,DEPART_TIME,
SEN_STATUS,SEN_ID) values ('A ',1122,to_timestamp('27-FEB-11 01.49.33.000000000 PM',
'DD-MON-RR HH.MI.SS.FF AM'),null,'1',1);
-------------------------------------------------
--   END DATA FOR TABLE SENSOR
-------------------------------------------------
--   DATA FOR TABLE SENSOR_DETAIL
--   FILTER = none used
-------------------------------------------------
REM INSERTING into SENSOR_DETAIL
-------------------------------------------------
--   END DATA FOR TABLE SENSOR_DETAIL
-------------------------------------------------
--   Constraints for Table SENSOR_DETAIL
-------------------------------------------------------
  ALTER TABLE "SENSOR_DETAIL" MODIFY ("SEN_ID" NOT NULL ENABLE);
```

```
-----------------------------------------------------
--   DDL for Trigger PARKING_FINISH
-----------------------------------------------------
  CREATE OR REPLACE TRIGGER "PARKING_FINISH"
  after update ON trigger_table
  For each row
Declare
  i number;
  Begin
  i:=:old.sen_id;
  -- the above value  must be come from sensor camera as snapshot --
   begin
     out_sensor(i);
   end;
   begin
     PARKING_COST_CALC(i);
    end;
  End;
/
ALTER TRIGGER "PARKING_FINISH" ENABLE;
-----------------------------------------------------
--  DDL for Trigger PARKING_INIT
-----------------------------------------------------
 CREATE OR REPLACE TRIGGER "PARKING_INIT"
 AFTER INSERT ON trigger_table
   For each row
Declare
 i number;
 vc sensor.VEHICLE_CHARACTERS%type;
 vn sensor.VEHICLE_NUMBER%type;
 Begin
 i:=:new.sen_id;
 vc:= :new.VEHICLE_CHARACTERS;
 vn:= :new.VEHICLE_NUMBER;
-- the above values must be come from sensor camera as snapshot --
  begin
    In_sensor(i,vc,vn);
   end;
 End;
/
ALTER TRIGGER "PARKING_INIT" ENABLE;
-----------------------------------------------------
--  DDL for Function TIME_RATIO
-----------------------------------------------------
  CREATE OR REPLACE FUNCTION "TIME_RATIO" (ttime IN NUMBER) RETURN number AS
    Ttcost number(5,1);
    vtime number;
    pvalue number;
 BEGIN
  if (ttime>=1 and ttime<4) then
     vtime:=1;
   elsif (ttime>=4 and ttime<=24) then
      vtime:=2;
   elsif ttime>24  then
      vtime:=3;
      pvalue:= ceil(ttime/24)*2 ;
  end if;
  SELECT DECODE((vtime), 1, ttime*0.3,2, 2,3,pvalue)
        INTO Ttcost FROM dual;
   RETURN Ttcost;
 END Time_Ratio;
/
-----------------------------------------------------
--  DDL for Package PARKING
-----------------------------------------------------
  CREATE OR REPLACE PACKAGE "PARKING" is i number;
procedure Parking_cost_calc(i sensor.SEN_ID%type) ;
PROCEDURE IN_SENSOR (i number,
  vc sensor.VEHICLE_CHARACTERS%type ,
  vn sensor.VEHICLE_NUMBER%type);
PROCEDURE  OUT_SENSOR (i number);
PROCEDURE  PARKING_RESET  (sid sensor.SEN_ID%type);
PROCEDURE "SEN_UPDATE" (i number);
PROCEDURE SNAP_SHOT;
PROCEDURE "TIME_INTERVAL" (sid sensor.SEN_ID%type);
FUNCTION Time_Ratio(ttime IN NUMBER) RETURN number;
end;
```

```
--------------------------------------------------------
--  DDL for Package Body PARKING
--------------------------------------------------------
  CREATE OR REPLACE PACKAGE BODY "PARKING" is
procedure Parking_cost_calc
(i sensor.SEN_ID%type) is
 ttime number(5);-- parking time--
 ttcost number;  -- total cost according to parking time--
 vsensor_id sensor.SEN_ID%type;
st sensor.start_time%type;
de sensor.depart_time%type;
vc sensor.VEHICLE_characters%type;
vn sensor.VEHICLE_number%type;
 begin
   select start_time, depart_time ,ceil((depart_time-start_time)*24), SEN_ID,
  VEHICLE_characters,VEHICLE_number into
   st,de,ttime,vsensor_id,vc,vn from sensor;
   select Time_Ratio(ttime)into ttcost from dual;
-- store total cost for each CAR(vehicl_characters and vehicl number and dura
   insert into parking_cost
values vc,vn,vsensor_id,ttcost,st,de);
-- when no car in sensor, its status must be ready--
-- both start and departure time must be reset--
   begin
     Parking_reset(vsensor_id);
   end;
   --commit;--
end;
PROCEDURE "IN_SENSOR" (i number,vc sensor.VEHICLE_CHARACTERS%type,
  vn sensor.VEHICLE_NUMBER%type) is
   begin
    insert into sensor(VEHICLE_CHARACTERS,VEHICLE_NUMBER,START_TIME,SEN_ID,sen_status)
    values  (upper(vc),vn,sysdate,i,1);
  end;
PROCEDURE "OUT_SENSOR" (i number) IS
    BEGIN
    update sensor set DEPART_TIME=SYSDATE,sen_status=0  WHERE SEN_ID=i;
  end;
PROCEDURE "PARKING_RESET" (sid sensor.SEN_ID%type)
  is
 begin
-- when no car in sensor, its status must be ready--
-- both start and departure time must be reset--
   update sensor set sen_status=0,start_time=null,
       depart_time=null,VEHICLE_CHARACTERS=null,
       VEHICLE_NUMBER=null
      where SEN_ID = sid;
   -- commit;--
end;
PROCEDURE "SEN_UPDATE" (i number)  is
    begin
     update trigger_table set  sen_id =i where sen_id=i;
     --commit;  --
   end;
PROCEDURE SNAP_SHOT is
    begin
     insert into TRIGGER_TABLE
      values  (4,'lk',876);
   commit;
   end;
PROCEDURE "TIME_INTERVAL" (sid sensor.SEN_ID%type) IS
   ttime number(5);-- parking time--
   ttcost number;  -- total cost according to parking time--
   st_time sensor.start_time%type;
   de_time sensor.depart_time%type;
   vsid sensor.SEN_ID%type;
   vc sensor.VEHICLE_CHARACTERS%type;
   vn sensor.vehicle_number%type;
   begin
    select start_time, depart_time ,ceil((depart_time-start_time)*24), SEN_ID,
     vehicle_characters,vehicle_number
    into
   st_time,de_time,ttime, vsid,vc,vn from sensor where sen_id=sid;
  select TGrade(ttime)into ttcost from dual;
```

```
-- store total cost for each car(character and number and duration time)--
   insert into parking_cost
     values
   (vc,vn,vsid,ttcost,st_time,de_time);
   commit;
end;
FUNCTION TGrade (ttime IN NUMBER) RETURN number AS
    Ttcost number(5,1);
    vtime number;
    pvalue number;
 BEGIN
  if (ttime>=1 and ttime<4) then
     vtime:=1;
   elsif (ttime>=4 and ttime<=24) then
      vtime:=2;
   elsif ttime>24  then
      vtime:=3;
      pvalue:= ceil(ttime/24)*2 ;
  end if;
  SELECT DECODE((vtime), 1, ttime*0.3,2, 2,3,pvalue)
        INTO Ttcost FROM dual;
   RETURN Ttcost;
 END TGrade;
End;
/
-----------------------------------------------------
--  DDL for Procedure IN_SENSOR
-----------------------------------------------------
set define off;
  CREATE OR REPLACE PROCEDURE "IN_SENSOR" (i number,vc sensor.VEHICLE_CHARACTERS%type,
  vn sensor.VEHICLE_NUMBER%type) is
   begin
    insert into sensor(VEHICLE_CHARACTERS,VEHICLE_NUMBER,START_TIME,SEN_ID,sen_status)
    values (upper(vc),vn,sysdate,i,1);
  end;
/
--  DDL for Procedure OUT_SENSOR
-----------------------------------------------------
set define off;
  CREATE OR REPLACE PROCEDURE "OUT_SENSOR" (i number) IS
    BEGIN
    update sensor set DEPART_TIME=SYSDATE,sen_status=0  WHERE SEN_ID=i;
  end;
/
-----------------------------------------------------
--  DDL for Procedure PARKING_COST_CALC
-----------------------------------------------------
set define off;
  CREATE OR REPLACE PROCEDURE "PARKING_COST_CALC"
 (i sensor.SEN_ID%type) is
 ttime number(5);-- parking time--
 ttcost number;  -- total cost according to parking time--
 vsensor_id sensor.SEN_ID%type;
st sensor.start_time%type;
de sensor.depart_time%type;
vc sensor.VEHICLE_characters%type;
vn sensor.VEHICLE_number%type;
 begin
   select start_time, depart_time ,ceil((depart_time-start_time)*24), SEN_ID,
  VEHICLE_characters,VEHICLE_number into
   st,de,ttime,vsensor_id,vc,vn from sensor where SEN_ID=i;
   select Time_Ratio(ttime)into ttcost from dual;
-- store total cost for each vehicle(vehicl_characters and vehicl number and dura
   insert into parking_cost
values  (vc,vn,vsensor_id,ttcost,st,de);
-- when no car in sensor, its status must be ready--
-- both start and departure time must be reset--
   begin
     Parking_reset(vsensor_id);
   end;
   --commit;--
end;
/
-----------------------------------------------------
```

```
--  DDL for Procedure PARKING_RESET
------------------------------------------------------
set define off;
  CREATE OR REPLACE PROCEDURE "PARKING_RESET" (sid sensor.SEN_ID%type)
  is
 begin
-- when no car in sensor, its status must be ready--
-- both start and departure time must be reset--
   update sensor set sen_status=0,start_time=null,
       depart_time=null,VEHICLE_CHARACTERS=null,
       VEHICLE_NUMBER=null
      where SEN_ID = sid;
   -- commit;--
end;
/
--  DDL for Procedure SEN_UPDATE
------------------------------------------------------
set define off;
  CREATE OR REPLACE PROCEDURE "SEN_UPDATE" is
    begin
     update trigger_table set  sen_id =11 where sen_id=11;
     --commit;  --
   end;
/
------------------------------------------------------
--  DDL for Procedure SNAP_SHOT
------------------------------------------------------
set define off;
  CREATE OR REPLACE PROCEDURE "SNAP_SHOT" is
    begin
     insert into TRIGGER_TABLE
      values  (1,'a',1122);
   commit;
   end;
/
------------------------------------------------------
--  DDL for Procedure TIME_INTERVAL
------------------------------------------------------
set define off;
  CREATE OR REPLACE PROCEDURE "TIME_INTERVAL" (sid sensor.SEN_ID%type) IS
   ttime number(5);-- parking time--
   ttcost number;  -- total cost according to parking time--
   st_time sensor.start_time%type;
   de_time sensor.depart_time%type;
   vsid sensor.SEN_ID%type;
   vc sensor.VEHICLE_CHARACTERS%type;
   vn sensor.vehicle_number%type;
   begin
    select start_time, depart_time ,ceil((depart_time-start_time)*24), SEN_ID,
     vehicle_characters,vehicle_number
    into
   st_time,de_time,ttime, vsid,vc,vn from sensor where sen_id=sid;
  select Time_ratio(ttime)into ttcost from dual;
-- store total cost for each vehicle(vehicl-character and vehicl number and duration time)--
   insert into parking_cost
     values
   (vc,vn,vsid,ttcost,st_time,de_time);
   commit;
end;
/
```