F.A. Schreiber, R. Camplani, M. Fortunato, M. Marelli

PERLA: A DECLARATIVE LANGUAGE AND MIDDLEWARE FOR PervasIVE SYSTEMS

3rd EuroSSC, Zurich, October 29-31, 2008
Adjunct Proceedings – Posters and Demo Abstracts
pp. 19-20
PERLA: A DECLARATIVE LANGUAGE AND MIDDLEWARE FOR PERVASIVE SYSTEMS

Authors F.A. Schreiber, R. Camplani, M. Fortunato, M. Marelli
{schreibe, camplani}@elet.polimi.it
Politecnico di Milano, Dipartimento di Elettronica e Informazione, Milano, Italy

Abstract. The system aims at managing, with a unique language, data sampled from different nodes of a pervasive system, providing them to the final user. It is composed of three elements: the nodes are heterogeneous devices equipped with sensors, which collect data and send them through the network managed by the middleware; PERLA (PERvasive LAnguage) is the SQL like language used to query logical objects, that are abstractions of the nodes; the middleware is a stack of software layers providing an implementation of the logical object abstraction. Answers to the queries take into account the context the system operates within, in order to provide the user with data appropriate to different environmental conditions.

1 The Language

PERLA is a language which allows the user to interact with logical objects by wrapping physical devices [1]. It is worth to notice that a logical object can abstract both a single sensor node and a set of devices (e.g. a whole WSN). The language currently supports three kinds of queries.

Low Level Queries are executed on a single logical object and define how and when the sampling should be performed, how sampled data should be locally processed, and which results have to be produced. A clause is provided by the syntax of these queries to specify the conditions defining the set of logical objects the query will be instantiated and executed on. A clause has been introduced to support a special operation that allows dynamic changes in the set of logical objects executing a specific query, based on the results produced by another running query. We call this operation PILOT JOIN, due to the conceptual similarity with standard SQL join operation. In fact, it forces each involved logical object to start (or stop) the query execution if the joined stream contains (or not) a record that matches the current value of logical object attributes, as specified in the condition part of the PILOT JOIN clause. This operation is the key feature enabling the execution of context dependent queries [2]: the content of the joined stream is a description of the current environmental situation, while the join condition defines the context-aware data tailoring the user is interested in. Moreover, the values of the current matching record can be used

* Work supported by: MIUR Art-Deco and Politecnico di Milano Prometeo projects
in other clauses to adapt the query behavior to the current context. **High Level Queries** are very similar to the normal streaming databases ones and they allow to manage different streams produced by low level queries. Finally, **Actuation Queries** are not intended to collect data from a logical object, but rather to send commands to the logical object they are executed on.

## 2 The Middleware

The goal of the middleware is to provide an abstraction for each device in terms of logical objects and to support the execution of PERLA queries. During the design of the middleware, we strove to make the definition and the addition of new devices easy. We also tried to minimize the amount of low level code the user has to write to make the new device recognizable by the middleware. When a query is submitted to the system, the **language parser** transforms it in a suitable format for distribution and execution. High level queries are executed by the **high level query executor** which is a streaming database engine. The process is more complex for low level queries: firstly the **logical objects registry** is used to find the set of devices that will be involved in the query execution. Then, each involved **low level query executor** starts sampling its device (through the abstraction provided by the logical object) and managing sampled data as required by the query. We decided to implement the whole software, from the logical objects layer up, using JAVA technology. We also assumed that each logical object (that we can now consider as a JAVA remote object) is reachable via TCP/IP. If the physical device is connected to a different network (e.g. a CAN-BUS channel), the correspondent logical object will be instantiated on the nearest device equipped with both a JAVA Virtual Machine and a TCP/IP connection. The communication protocol between the logical object and the physical device is managed by a specific software layer provided with the middleware. Another important component is a **C library** that has been developed to minimize the low level programming effort by the user to integrate a new technology in the middleware. To reach this goal we defined the structure of a **XML file**, containing a full description of a device in terms of available sensors, measures that can be sampled and packets format. The effort required to the user to integrate a new technology is limited to the creation of the XML descriptor and the extension of the C library with the definition of device specific sampling routines. The logical object wrapping the device is then automatically and dynamically generated by the middleware.

### References

FULL DEclarative SQL-Like HIGH LEVEL LANGUAGE to query
PERVASIVE SYSTEMS
hiding the complexity of handling
DIFFERENT TECHNOLOGIES

DATA MANAGEMENT SECTION

Data Type

WSN

Adapter Cln

FPC 2

HLD

Device 2

THE MIDDLEWARE:

• THE LANGUAGE SEMANTICS IS DEFINED ON THE CONCEPT OF LOGICAL OBJECT

• EACH DEVICE IS ABSTRACTED AS A LOGICAL OBJECT:

  • ATTRIBUTES (id, temperature, pressure, level, last sensed RFID reader, …)

  • EVENTS (last sensed RFID reader changed, …)

  • META-DESCRIPTION (name, data type, … for each attribute)

THE GOALS OF THE MIDDLEWARE ARE:

• TO PROVIDE AN ABSTRACTION FOR EACH DEVICE IN TERMS OF LOGICAL OBJECTS

• TO SUPPORT THE EXECUTION OF PERLA QUERIES MAKING THE DEFINITION AND THE ADDITION OF THE NEW DEVICES (AND NEW TECHNOLOGIES) EASY

THE MIDDLEWARE:

• PARSES AND VERIFIES PERLA QUERIES

• DYNAMICALY GENERATES LOGICAL OBJECTS CORRESPONDENT TO PHYSICAL DEVICES

• MANAGES THE COMMUNICATION PROTOCOL BETWEEN PHYSICAL DEVICES AND LOGICAL OBJECTS

• SENDS HIGH LEVEL QUERIES TO THE HIGH LEVEL EXECUTOR AND LOW LEVEL QUERIES TO LOGICAL OBJECTS

• THE EFFORT REQUIRED TO THE USER TO INTEGRATE A NEW TECHNOLOGY IS LIMTED TO:

  • THE CREATION OF AN XML DESCRIPTOR

  • THE DEFINITION OF DEVICE SPECIFIC SAMPLING Routines

THE MIDDLEWARE ARCHITECTURE:

• RUN-TIME SUPPORT OF HETEROGENEITY:

  • WSN nodes

  • RFID tags

  • PDAs

  • AD HOC boards

  • …

• SUPPORT OF NON INTELLIGENT DEVICES (RFID TAGs can be queried exactly as WSN nodes)

LOGICAL OBJECTS

• THE CREATIoN OF AN OBJECT

• THE CREATION OF AN OBJECT

• THE CREATION OF A QUERY

• THE CREATION OF A QUERy

• THE CREATION OF AN ACTUATION QUERY

• THE CREATION OF A TARGET

• THE CREATION OF A TASK

• THE CREATION OF A TASK

• THE CREATION OF A TASK

• …

THE PILOT JOIN operation

• PILOT JOIN is a special operation that allows dynamic changes in the set of logical objects executing a specific low level query, based on the results produced by another running query.

• It forces each involved logical object to start (or stop) the query execution if the joined stream contains (or not) a record that matches the current value of logical object attributes, as specified in the condition part of the PILOT JOIN clause.

• The PILOT JOIN operation is the key feature enabling the execution of CONTEXT DEPENDENT QUERIES:

  • the context of the joined stream is a description of the current ENVIRONMENTAL SITUATION.

  • the join condition defines the CONTEXT-AWARE DATA TAILORING the user is interested in.

A COMPLETE QUERY EXAMPLE

There is a set of tanks, containing some temperature sensors. A GPS and a base station are mounted on each tank.

The temperature sensors contained in the tank nearest to a given point P must be sampled every minute.

The distances of the tanks from the point P must be revaluated every hour.

The GPS node is a logical object wrapping an adapter that connects to a GPS device.

The PILOT JOIN operation

• PILOT JOIN is a special operation that allows dynamic changes in the set of logical objects executing a specific low level query, based on the results produced by another running query.

• It forces each involved logical object to start (or stop) the query execution if the joined stream contains (or not) a record that matches the current value of logical object attributes, as specified in the condition part of the PILOT JOIN clause.

• The PILOT JOIN operation is the key feature enabling the execution of CONTEXT DEPENDENT QUERIES:

  • the context of the joined stream is a description of the current ENVIRONMENTAL SITUATION.

  • the join condition defines the CONTEXT-AWARE DATA TAILORING the user is interested in.

FIRST PERLA DEPLOYMENT: ROCKFALL MONITORING on San Martino face (Lecco, Italy)

• ACCELEROMETERS AND GEOFENCES ARE USED TO MONITOR INCLINATION AND VIBRATIONS

• NODES ARE DAPC BOARDS (C BASED) ABLE TO DETECT EVENTS ANALYZING ACCELEROMETERS SIGNALS

• LOCAL COORDINATORS ARE AD HOC LINUX BOARDS (JAVA BASED), ABLE TO HOST LOGICAL OBJECTS

• THE CHANNEL BETWEEN NODES AND LOGICAL OBJECTS IS A CAN-BUS

• PERLA PARSER IS LOCATED ON A PC IN THE REMOTE CONTROL CENTER

THE TARGETS

• RUN-TIME SUPPORT OF HETEROGENEITY:

  • WSN nodes

  • RFID tags

  • PDAs

  • AD HOC boards

  • …

• SUPPORT OF NON INTELLIGENT DEVICES (RFID TAGs can be queried exactly as WSN nodes)

THE MIDDLEWARE:

• THE LANGUAGE SEMANTICS IS DEFINED ON THE CONCEPT OF LOGICAL OBJECT

• EACH DEVICE IS ABSTRACTED AS A LOGICAL OBJECT:

  • ATTRIBUTES (id, temperature, pressure, level, last sensed RFID reader, …)

  • EVENTS (last sensed RFID reader changed, …)

  • META-DESCRIPTION (name, data type, … for each attribute)

THE GOALS OF THE MIDDLEWARE ARE:

• TO PROVIDE AN ABSTRACTION FOR EACH DEVICE IN TERMS OF LOGICAL OBJECTS

• TO SUPPORT THE EXECUTION OF PERLA QUERIES MAKING THE DEFINITION AND THE ADDITION OF THE NEW DEVICES (AND NEW TECHNOLOGIES) EASY

THE MIDDLEWARE:

• PARSES AND VERIFIES PERLA QUERIES

• DYNAMICALY GENERATES LOGICAL OBJECTS CORRESPONDENT TO PHYSICAL DEVICES

• MANAGES THE COMMUNICATION PROTOCOL BETWEEN PHYSICAL DEVICES AND LOGICAL OBJECTS

• SENDS HIGH LEVEL QUERIES TO THE HIGH LEVEL EXECUTOR AND LOW LEVEL QUERIES TO LOGICAL OBJECTS

• THE EFFORT REQUIRED TO THE USER TO INTEGRATE A NEW TECHNOLOGY IS LIMTED TO:

  • THE CREATION OF AN XML DESCRIPTOR

  • THE DEFINITION OF DEVICE SPECIFIC SAMPLING Routines

THE MIDDLEWARE ARCHITECTURE:

• RUN-TIME SUPPORT OF HETEROGENEITY:

  • WSN nodes

  • RFID tags

  • PDAs

  • AD HOC boards

  • …

• SUPPORT OF NON INTELLIGENT DEVICES (RFID TAGs can be queried exactly as WSN nodes)

LOGICAL OBJECTS

• THE CREATIoN OF AN OBJECT

• THE CREATION OF AN OBJECT

• THE CREATION OF A QUERY

• THE CREATION OF A QUERy

• THE CREATION OF AN ACTUATION QUERY

• THE CREATION OF A TARGET

• THE CREATION OF A TASK

• THE CREATION OF A TASK

• THE CREATION OF A TASK

• …

THE PILOT JOIN operation

• PILOT JOIN is a special operation that allows dynamic changes in the set of logical objects executing a specific low level query, based on the results produced by another running query.

• It forces each involved logical object to start (or stop) the query execution if the joined stream contains (or not) a record that matches the current value of logical object attributes, as specified in the condition part of the PILOT JOIN clause.

• The PILOT JOIN operation is the key feature enabling the execution of CONTEXT DEPENDENT QUERIES:

  • the context of the joined stream is a description of the current ENVIRONMENTAL SITUATION.

  • the join condition defines the CONTEXT-AWARE DATA TAILORING the user is interested in.

A COMPLETE QUERY EXAMPLE

There is a set of tanks, containing some temperature sensors. A GPS and a base station are mounted on each tank.

The temperature sensors contained in the tank nearest to a given point P must be sampled every minute.

The distances of the tanks from the point P must be revaluated every hour.

The GPS node is a logical object wrapping an adapter that connects to a GPS device.

The PILOT JOIN operation

• PILOT JOIN is a special operation that allows dynamic changes in the set of logical objects executing a specific low level query, based on the results produced by another running query.

• It forces each involved logical object to start (or stop) the query execution if the joined stream contains (or not) a record that matches the current value of logical object attributes, as specified in the condition part of the PILOT JOIN clause.

• The PILOT JOIN operation is the key feature enabling the execution of CONTEXT DEPENDENT QUERIES:

  • the context of the joined stream is a description of the current ENVIRONMENTAL SITUATION.

  • the join condition defines the CONTEXT-AWARE DATA TAILORING the user is interested in.