DESIGN OF A DECLARATIVE LANGUAGE FOR PERVASIVE SYSTEMS

22 luglio 2008

Tesi di Laurea Specialistica:

Marco Fortunato
Marco Marelli

Relatore:

Prof. Fabio A. Schreiber

Co-relatore:

Ing. Romolo Camplani

Introduction

DESIGN

of a

DECLARATIVE DATA LANGUAGE

for

PERVASIVE SYSTEMS

+

DEVELOPMENT

of a

MIDDLEWARE

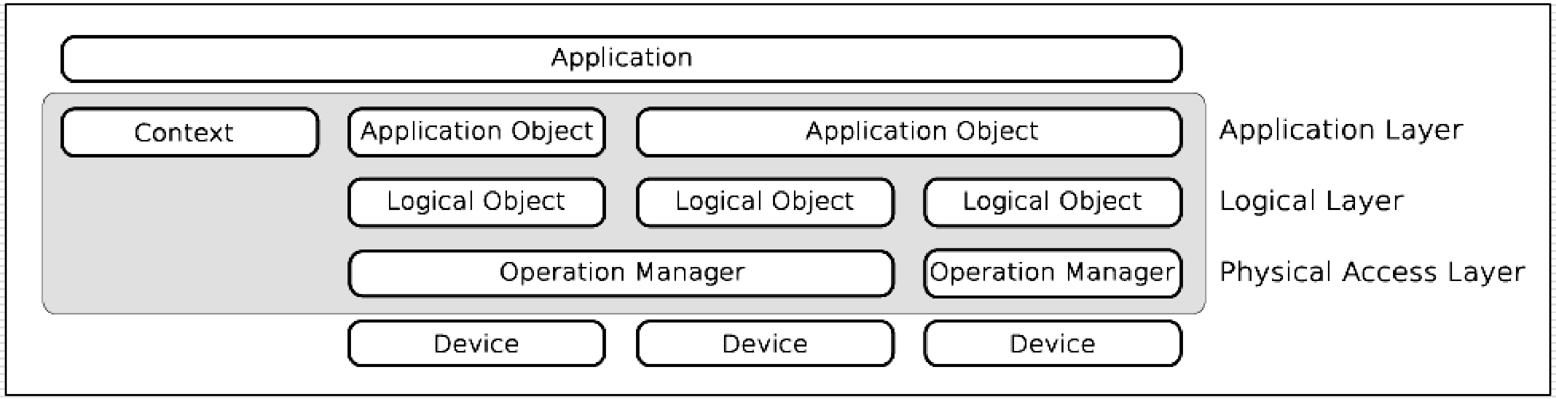
- □ **WSN** *(Wireless Sensor Network)*
- □ **RFID** *(Radio Frequency Identification)*
- □ **PDA**
- □ **…**

Goals of the projects

		Deploy-time heterogeneity	
		Little	**Full**
Run-time heterogeneity	**Little**	Homogeneous systems	Partially Homogeneous systems
	Full	-	***Heterogeneous systems***

- ***Network abstraction***
  - Provide a database view of the network
  - Hide low level characteristics of physical devices
- ***Functional features***
  - Set sampling parameters (e.g. sample rate)
  - Manipulate sampled data to produce query results
- ***Non functional features***
  - Many non functional constraints can be identified:
    - to decide if a node should participate to a query
    - to set the rate used for sending data out of the node
    - to retrieve information about network nodes

Architecture for pervasive systems



- □ *Application Layer*: Front end for data access
- □ *Logical Object Layer*: Abstraction for physical devices
- □ *Device Access Layer*: Sw infrastructure for device access

Logical objects (1)

- Logical object functionalities
  - ***Retrieve attributes***
    - ***STATIC*** attributes
      - *ID, device_type, maximum_sampling_rate, location (fixed devices)*
    - ***PROBING DYNAMIC*** attributes
      - *temperature, pressure, location (mobile devices), power_level*
    - ***NON PROBING DYNAMIC*** attributes
      - *last_sensed_RFID_reader*
  - ***Fire notification events***
  - ***Get the list of supported attributes and events***

Logical objects (2)

- Abstraction of the sampling operation:
  - ***PERIODIC SAMPLING***
    - Reading of a logical object attribute periodically
  - ***EVENT BASED SAMPLING***
    - Reading of a logical object attribute after an event is raised

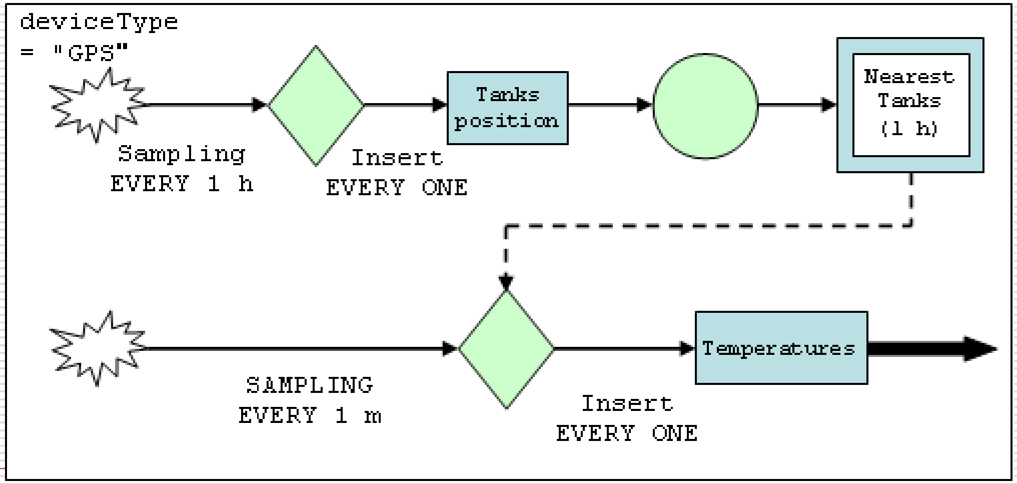
Language design

```
CREATE STREAM TanksPositions (gpsID ID, linkedBaseStationID ID, distanceFromP FLOAT) AS
LOW:
  EVERY ONE
  SELECT ID, linkedBaseStationID, dist_from_P(locationX, locationY)
  SAMPLING EVERY 1 h
  EXECUTE IF deviceType = "GPS"


CREATE SNAPSHOT NearestTank (gpsID ID, linkedBaseStationID ID)
WITH DURATION 1 h AS
HIGH:
  SELECT TanksPositions.gpsID, TanksPositions.linkedBaseStationID
  FROM TanksPositions (1 h)
  WHERE TanksPositions.distanceFromP = MIN(TanksPositions.distanceFromP)


CREATE OUTPUT STREAM Temperatures (sensorID ID, temp FLOAT) AS
LOW:
  EVERY ONE
  SELECT ID, temp
  SAMPLING EVERY 1 m
  PILOT JOIN NearestTank ON NearestTank.linkedBaseStationID = baseStationID
```

- User submitted query
  - Data structures
    - *Streams*
    - *Snapshots*
  - *Queries*
    - *Low level queries*
    - *High level queries*



Pilot join (1)

- **The *PILOT JOIN* operation activates the execution of a low level query on logical objects conditioned by values sampled on *OTHER NODES***

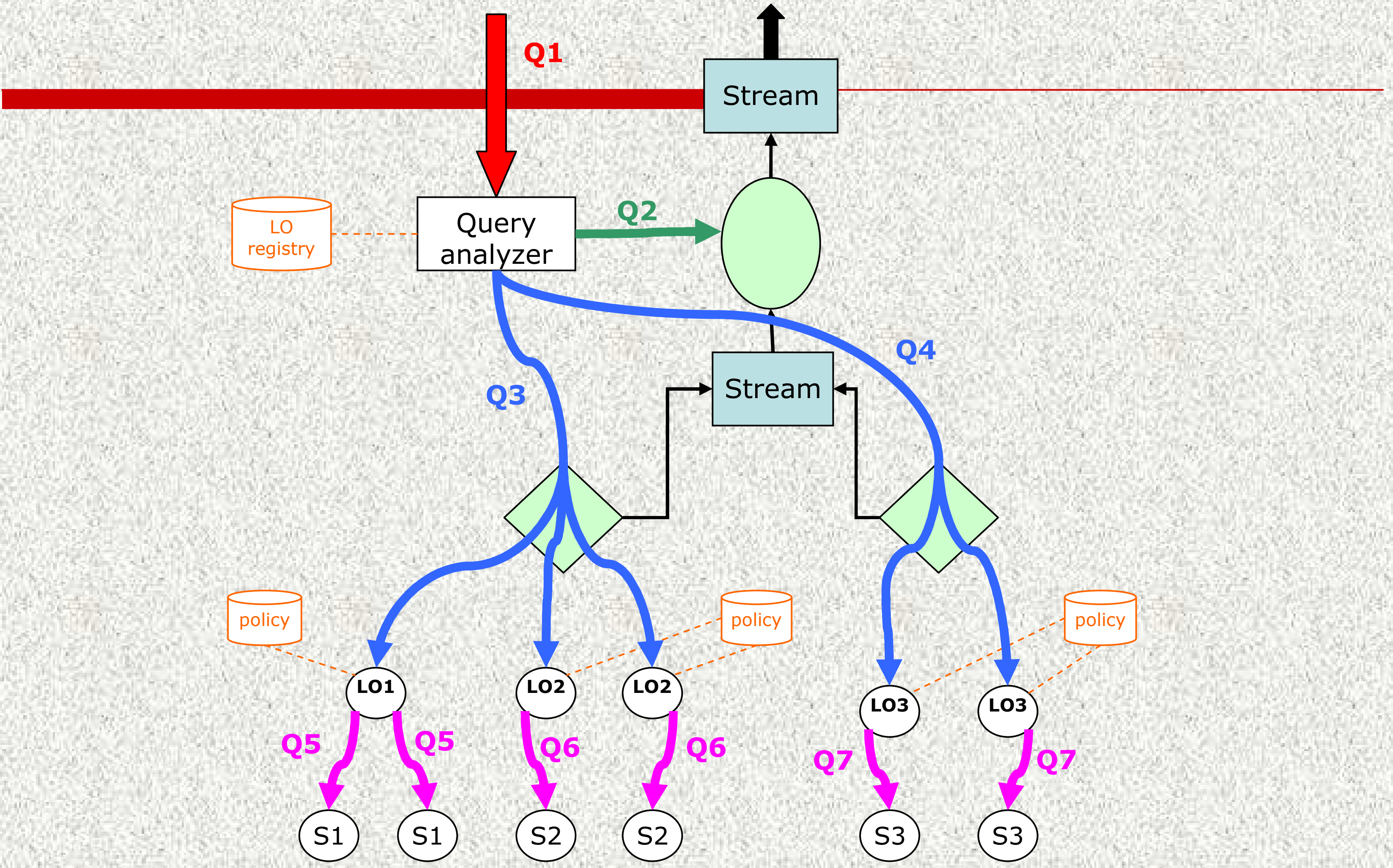
- Example:
  - Monitor the temperature of all the pallets in trucks whose current position is in a given parking area
    - Temperature sensors on pallets
    - Position sensors on trucks

```
PILOT JOIN BaseStationList ON
           currentBaseStation = baseStationList.baseStationID
```

Pilot join (2)

- Two types of pilot join are supported:
  - ***EVENT BASED*** pilot join
    - ***When an event happens, a given set of nodes are fired to sample*** ***(***e.g.: sense pallet temperature for 15 minutes every time a truck enters parking area *B)*
  - ***CONDITION BASED*** pilot join
    - ***Continuous sampling is performed on nodes connected to a given base station*** (e.g.: start sampling every 15 minutes the temperature of pallets whose last sensed position was in parking area *B)*

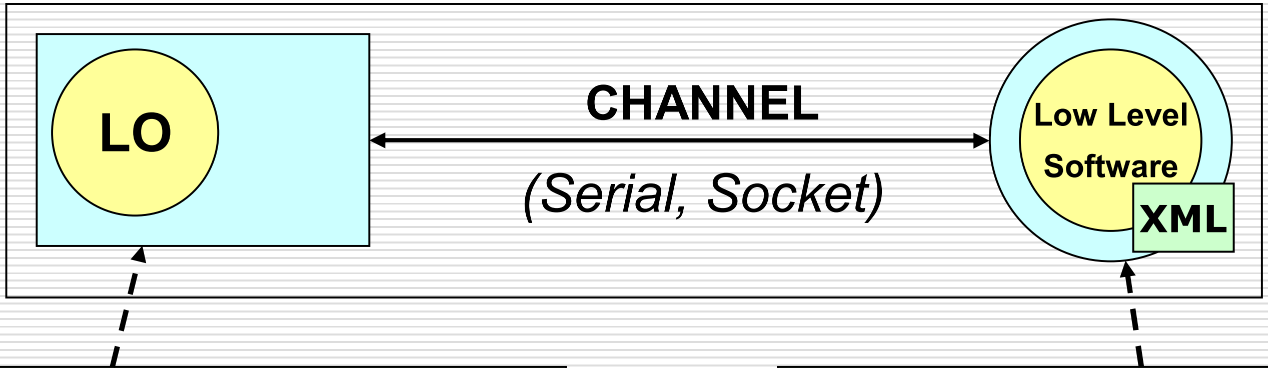
Query execution



Middleware (1 / 3)

- How logical objects (and registry) should be implemented?
- Goals of the middleware:
  - Supporting query execution
  - Making the addition of new technologies easy
- Two possible approaches:
  - Completely distributed implementation (more research oriented)
  - Partially distributed implementation (rapid prototype development)

Middleware (2 / 3)

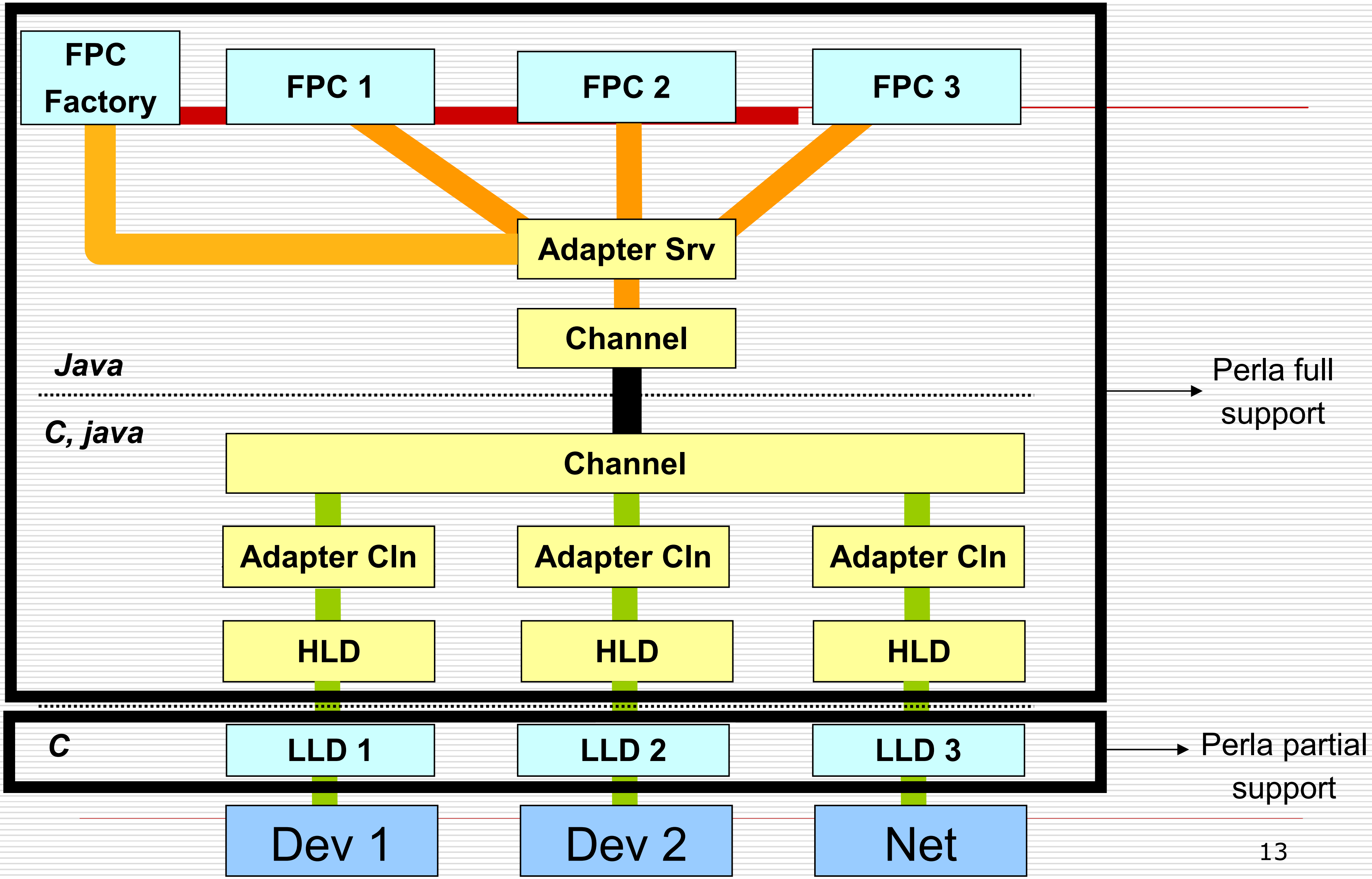


- Linux board or PC
- Java virtual machine
- TCP-IP connection to middleware network

- Acquisition unit
- Simple node (microcontroller, …)
- C based

- LO can be hosted on the acquisition unit, if it is powerful enough

Middleware (3 / 3)



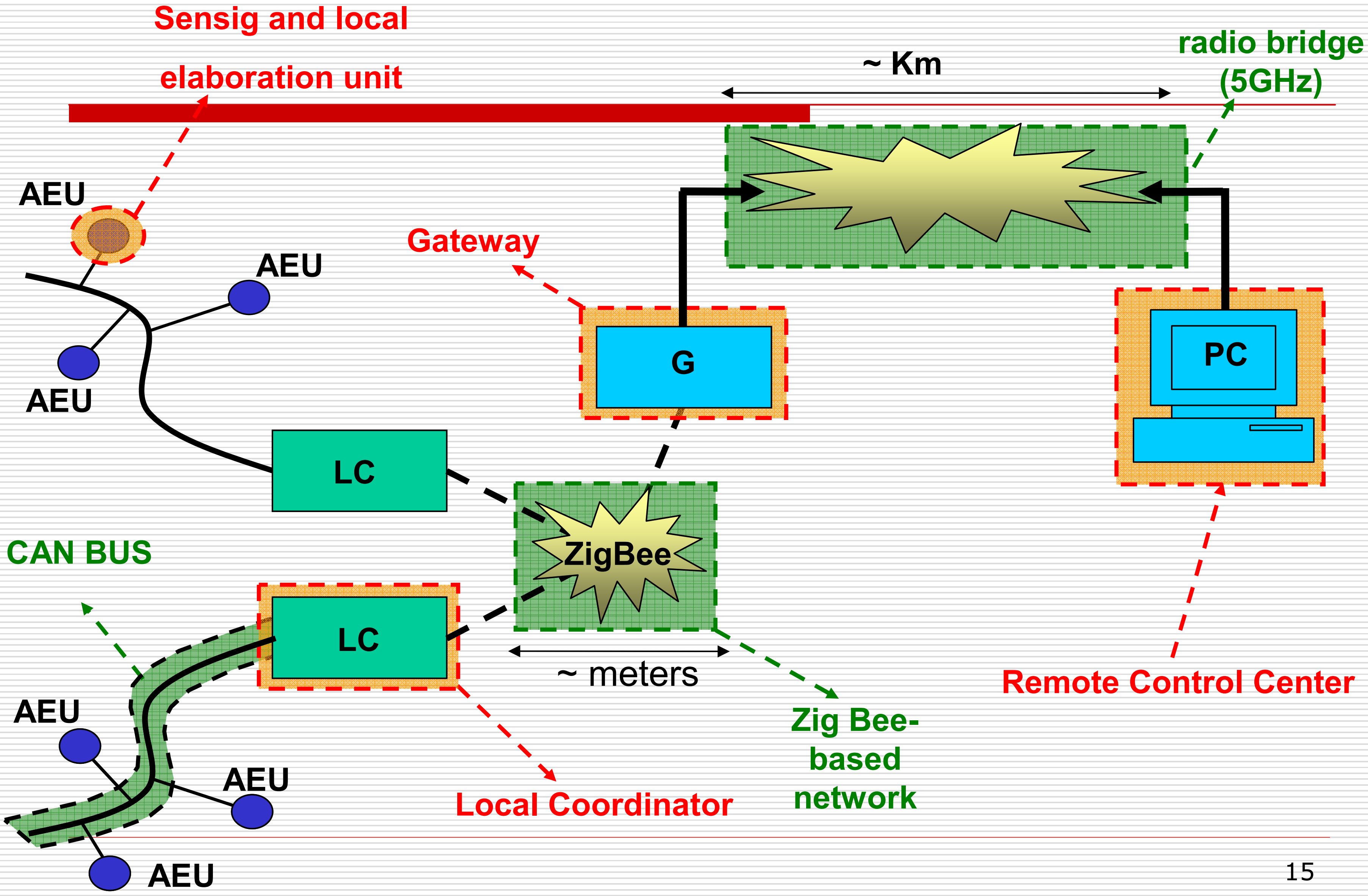
PERLA – PERvasive LAnguage

- Fabio A. Schreiber, Romolo Camplani, Marco Fortunato, Marco Marelli,Filippo Pacifici:

  "PERLA: a Data Language for Pervasive Systems"

  in Proceedings of Sixth Annual IEEE International Conference on Pervasive Computing and Communications (PerCom 2008). Honk Kong, pp. 282-287,2008.

PERLA real deployment: Rockfall monitoring



SOON IN SAN MARTINO FACE

THANKS FOR YOUR ATTENTION