EnviroSuite: An Environmentally Immersive Programming Framework for Sensor Networks

Liqian Luo, Tarek F. Abdelzaher | University of Illinois at Urbana-Champaign
Tian He | University of Minnesota
John A. Stankovic | University of Virginia

Presented by Mark Wood
November 27, 2006

CSE 521S Wireless Sensor Networks | Washington University in St. Louis

What is EnviroSuite?
- A new paradigm: Environmentally immersive programming (EIP)
- A programming framework to simplify sensor network programming
- An object-based programming system
- A middleware service for monitoring and tracking applications
- A software development suite, which includes a library (EIPLib) and a compiler (EIPLC)

EnviroSuite Objects
- **Environmental objects**
  - A sensory or geographic signature that defines an area
  - Data variables to be collected in an area
  - Methods that can be performed in its context
- **Function objects**
  - Regular objects which do not represent environmental elements

Software
- **EIPLib**
  - A programming library (in nesC) that provides the detailed implementations
  - Abstracts some of the low-level functions from the programmer
  - Simplifies programming for the user
- **EIPLC**
  - A compiler (in PERL) that translates EnviroSuite applications into nesC code

EnviroSuite: An Environmentally Immersive Programming Framework for Sensor Networks

Liqian Luo, Tarek F. Abdelzaher | University of Illinois at Urbana-Champaign
Tian He | University of Minnesota
John A. Stankovic | University of Virginia

Presented by Mark Wood
November 27, 2006

CSE 521S Wireless Sensor Networks | Washington University in St. Louis

What is EnviroSuite?
- A new paradigm: Environmentally immersive programming (EIP)
- A programming framework to simplify sensor network programming
- An object-based programming system
- A middleware service for monitoring and tracking applications
- A software development suite, which includes a library (EIPLib) and a compiler (EIPLC)

EnviroSuite Objects
- **Environmental objects**
  - A sensory or geographic signature that defines an area
  - Data variables to be collected in an area
  - Methods that can be performed in its context
- **Function objects**
  - Regular objects which do not represent environmental elements

Software
- **EIPLib**
  - A programming library (in nesC) that provides the detailed implementations
  - Abstracts some of the low-level functions from the programmer
  - Simplifies programming for the user
- **EIPLC**
  - A compiler (in PERL) that translates EnviroSuite applications into nesC code
Relation to nesC

Monitoring and Tracking Applications in EnviroSuite

EnviroSuite Middleware

EIPLC

- Primitive Algorithms
- High-level services

Monitoring and Tracking Applications in nesC

Object IDs

- Label a contiguous group of nodes that match a sensory signature
- Data operations can be performed on different locales based on their object ID
- Group state is stored in variables encapsulated within named objects

Object Declarations

- **Object type** - programmer-defined label
- **Object context** – mapping of the object to an environmental element
- **Object attribute** – measurements collected and aggregated in the object context
- **Object method** – encapsulated methods that perform computation, communication or actuation

Object Declaration - Example

```
object type
object context
object attribute
object method

object_condition = ferrous_object() & vehicle_sound();
```

Object Context

- EIPLib contains sensor data processing algorithms that can return:
  - sensor output, i.e. \texttt{temperature()}
  - environmental conditions, i.e. \texttt{vehicle\_sound()}
  - node attributes, i.e. \texttt{voltage()}

- Example:
  \texttt{object\_condition = altitude()>500 \&\& temperature()<32;}

Object Context

- Object conditions can also be parameterized.

- Example:
  \texttt{object\_condition = altitude()>500 \&\& temperature()<32;}

Object Context

- EIPLib contains sensor data processing algorithms that can return:
  - sensor output, i.e. \texttt{temperature()}
  - environmental conditions, i.e. \texttt{vehicle\_sound()}
  - node attributes, i.e. \texttt{voltage()}

- Example:
  \texttt{object\_condition = ferrous_object() \&\& vehicle\_sound();}
Object Attributes

- Sensor measurements and aggregation method
- Degree of confidence in received aggregation
- Freshness of attribute

```
object_attribute location
  attribute_value = AVERAGE(position())
  attribute_degree = 2
  attribute_freshness = 500ms
```

Object Methods

- Methods in `object_main_function` statements get called during object creation

```
object_main_function = vehicle.getLocation;
```

- Methods in `object_function` statements get called only by other objects

```
object_function = monitor.reportLocation;
```

Inter Object Call (IOC)

- Methods can use EnviroSuite communications primitives

To execute an IOC and declare its handler:

```
ES_IOC report = call monitorInstance.
  monitor.reportLocation(currentLocation);
```

To receive IOC result interrupts:

```
ES_IOCRESULT report(bool result) {
  // deal with remote call results here
  return;
}
```

Global Variables

- Globally shared static variables can be defined with `ES_READ` and `ES_WRITE`

```
command result_t vehicle.getLocation() {
  call ES_WRITE(monitorInstance.vehicleNumber,
    monitorInstance.vehicleNumber + 1);
  return call Timer.start(TIMER_REPEAT,500);}
```

EnviroSuite supports 3 types of objects:

- **Event objects** – created for mobile events that dynamically change their geographic location
- **Region objects** – mapped to static or slowly moving regions
- **Function objects** – not mapped to an environmental element

Event Object Maintenance

- Event objects are created dynamically
- **Multi-target group management protocol (MGMP)** gives the object a unique object ID
- Movement of the contiguous region changes group membership
- MGMP maintains the object ID for the event object despite mobility
- This is transparent to the programmer
State of Nodes Around an Environmental Event

[Diagram showing the state of nodes around an environmental event with roles such as leader, members, followers, and nulls.]

Maintaining Object Uniqueness
- Internally, MGMP elects a leader to maintain a unique object ID.
- The leader sends periodic heartbeats to nodes within the object resolution.
- The heartbeat period can affect the object creation latency.
- Nodes can have only one object ID.
- MGMP maintains a state machine for node state transitions.
- A delayed object creation mechanism is used to ensure that the group has elected a single leader.

Tracking Moving Events
- **Object migration** - Leadership handover, used in tracking moving events.
- **Follower** nodes hear heartbeats from leaders and know in advance the incoming event.
- When the event is sensed, the node joins existing objects as a member, instead of creating spurious objects.
- The trajectory of the leader is stored in its representing object.
- When 2 events cross, the leaders use a temporary object ID which is a concatenation of both, and track each other’s trajectory until a disambiguation algorithm can be used.

Region Object Maintenance
- Region objects are associated with a relatively fixed set of nodes.
- A default leader sends out tree requests.
- Spanning tree algorithm is used to construct a multi-parent diffusion tree with the leader as root.
- The tree’s outer boundary is defined by outer nodes.
- Adding and pruning of tree leaves is done to satisfy the object condition.
- Object attributes get aggregated along intermediate hops.
- The root of the aggregate tree is migrated to the location that minimizes communication and aggregation overhead, and to a higher energy node.

Evaluation: Object Creation Delay

[Graph showing object creation delay over heartbeat period with candidate periods.

Evaluation: Object Migration

[Graph showing maximum heartbeat reach (sink) depth over heartbeat period with half object resolutions.]
Evaluation: Object Crossing

Evaluation: Inter-Object Communication

Evaluation: Code Length

Critiques

- Hard-coded content (fixed number and type of objects you can track)
- The mote cannot access the object ID
- The object ID is an abstraction on the base station (not simplifying the code in the network itself)
- Requires a leader (electing a leader and changing the leader)

Summary

- Environmentally immersive programming can be used to abstract environmental elements
- EnviroSuite provides a library of algorithms for tracking and environmental monitoring
- Programming is simplified and code length is reduced
- Evaluation shows EnviroSuite performs well in maintaining object uniqueness and identity, and inter-object communication


Both have similar goals:
- Programming framework to simplify sensor network programming
- Middleware service for monitoring and tracking applications
- Confidence level for environmental element classification
- To abstract environmental elements, simplifying programming
- Network-based, rather than node-based language
- Energy-balancing in sensor networks

This work makes it possible for non-technical people to write efficient programs for sensor networks.
Questions?