Opportunistic Application Flows in Sensor-based Pervasive Environments

N. Jiang, C. Schmidt, V. Matossian, and M. Parashar
WINLAB/TASSL
ECE, Rutgers University
http://www.caip.rutgers.edu/TASSL

Presented by Nanyan Jiang
Emerging Pervasive Applications

- Peers (sensors, actuators, services, resources, …) opportunistically interact, coordinate and collaborate to satisfy goals
  - Scenarios (environmental monitoring, crisis management, emergency medical care, automated control, homeland security, …)
  - For example, a fire management application uses streaming information from sensors embedded in the building along with real time information (temperature, wind speed and direction, smoke density) to predict the spread of the fire and guide firefighter, warning of potential threats (blowback if door is open) and indicating most effective options
Sensor-based Pervasive Environments

- **Ad hoc structures/behaviors**
  - **hierarchical architecture**
    - dynamic sensors groups, elected FN, connected AP
- **Dynamic**
  - sensor join, leave, move, change behavior
- **Unreliable**
  - components, communication
- **Lack of common/complete knowledge**
  - sensor number, type, location, availability, connectivity, etc.
- **Heterogeneous**
  - capability, connectivity, reliability, guarantees, QoS
Project Meteor: Enabling Pervasive Applications

- Objective: enable pervasive applications based on symbiotically and opportunistically orchestrated interactions and coordination among sensors, resources, services, applications and data to satisfy dynamically defined objectives and goals
  - Sensors/resource/data/information/service discovery
  - Interactions and communications
    - content-based decentralized information exchange
    - multiple interaction semantics
    - manage system scale, dynamism, unreliability, …
  - Composition, coordination, workflows
    - global behavior without the need for global knowledge
    - opportunistic and emergent
- Programming model, interaction & coordination paradigm, middleware architecture
Overview: Opportunistic Application Flows

- **Associative Rendezvous (AR)**: An abstraction for content-based decoupled interaction with programmable reactive behaviors.

- **Cascading Local Behaviors (CLB)**: A programming model, where the behaviors of individual application elements are locally defined in terms of local state, and context and content events, and result in data and interest messages being produced.

- **Opportunistic Flows**: Application flows emerged as a consequence of the cascading effect of local behaviors, without having to be explicitly programmed.
Associative Rendezvous: Concepts

- Content-based decoupled interaction
  - Based on content rather than names and/or addresses
  - Supports asynchronous and decoupled interactions rather than forcing synchronizations
- Programmable reactive behaviors
  - Self-managing
  - Enable multicast, anycast...

Semantics of AR

- AR message
  - (header<profiles>, action, data)
- Symmetric post primitive
  - does not differentiate between interest/data
- Associative selection
  - keywords, partial keywords, wildcards, ranges...
- Actions
  - store, retrieve, notify, delete
Associative Rendezvous: An Illustrative Example

AR: Associative Rendezvous

\[ \text{post} \left( \langle p_1, p_2 \rangle, \text{notify\_interest}(C1) \right) \]

1. C1
2. \text{post} \left( \langle p_1, *> \right), \text{notify\_data}(C2) \)
3. notify\_interest(C1)
**AR: Associative Rendezvous**

- **Post:** \((p_1, p_2), \text{store}, \text{data})\)

- **Notify:** \(<p_1, *> \rightarrow \text{notify\_data}(C2)\)

- **Notify:** \(\text{notify}(C2)\)

Diagram:

- C1
- C2
- \((4)\)
- \((5)\)
AR: Associative Rendezvous

\[
\langle p_1, p_2 \rangle \rightarrow \text{store} \rightarrow \text{data}
\]

\[
\text{post}(\langle p_1, * \rangle, \text{retrieve} (C2))
\]

\[
\text{retrieve}(C2, \text{data})
\]
AR: Associative Rendezvous

post \((<p_1, p_2>, \text{delete\_data\ (C1)})\)  

\(<p_1, p_2> \rightarrow \text{store} \rightarrow \text{data}\)  

\(<p_1, *> \rightarrow \text{retrieve\ (C2)}\)  

post\((<p_1, *>, \text{delete\_interest\ (C2)})\)  

\(C1\)  

\(C2\)
Cascading Local Behaviors (CLB)

- Pervasive applications as emergent *opportunistic* flows
  - Local behavior of pervasive element defined as context and content based state transitions and interest/data messages
    - Content/context messages trigger local actions and/or generation of data/interest message
  - Cascading of local content/context-based transitions result in opportunistic application flows
An Illustrative Example

Temperature Sensor

if temp > 80
then publish temp

Thermostat

If temp > 85
Then temp_control = on;
publish temp_control

Window Actuator

If temp_control == on
then close windows

If temp > 85
post(temp > 85, notify_data)

notify(thermostat)

Turn temp_control on
post(temp_control, store)

Close window

post(temp = 91, store)

post(t*control, retrieve_data)

retrieve(temp_control)

if temp > 80
then publish temp
Meteor Programming Framework and Content-based Middleware

- **Self-organizing overlay**
  - **Rendezvous points**: Chord, a ring overlay
- **Content-based routing**
  - **Squid**
- **AR Messaging substrate**
  - **Profile manager**
  - **Matching engine**
- **CLB-based opportunistic flows**
  - **Local state machine**
  - **Message dispatcher**

![Diagram]

**Pervasive Application**

**(Opportunistic Flows)**

**Cascading Local Behaviors**

**Associative Rendezvous Messaging**

**Content-based Routing, Discovery**

**Self-organizing Overlay**

**Wireless/Wired substrate**

e.g. Sensors, actuators, internet
Experimental Evaluation (I)

- **Current Deployment**
  - *Linux-based sensor-network emulation testbed*
    - 64 nodes simulate RP and execute the Meteor stack
  - *Builds on JXTA p2p infrastructure*

- **Experimental evaluation:**
  - *Peer lookup overhead as a function of system size*
  - *Content-based routing overhead*
  - *Associative messaging overhead at each RP node.*
    - overhead of querying the database – associative selection
    - constructing of notification message – reactive behavior
Experimental Results (I)

Overlay network lookup overhead (Chord)

Content-based routing overhead (Squid)
Evaluation Results (II)

Matching overhead at a single RP (ARMS)
Conclusion

- Opportunistic content-based programming models and middleware infrastructures are required to address the challenges of pervasive applications.
- Associative rendezvous provides an abstraction for content-based decoupled interactions.
- Cascading local behaviors build on associative rendezvous to enable opportunistic application flows where flows emerge as a result of context and content based local behaviors.

More Information:
- Project page:
  - [http://www.caip.rutgers.edu/TASSL/Projects/Meteor/](http://www.caip.rutgers.edu/TASSL/Projects/Meteor/)
- Recent report:
  - [http://www.jxta.org/universities/rutgers.html](http://www.jxta.org/universities/rutgers.html)