Servilla: Service Provisioning in Wireless Sensor Networks

Chenyang Lu
Sensor Network Challenges

- Device heterogeneity
- Network dynamics
  - due to mobility and interference
- Limited resources
  - e.g., memory and energy

Structural Health Monitoring

- >26% of bridges were structurally deficient in 2009 [ASCE]
- Locates damage by analyzing a structure’s natural frequencies.
- Evaluated on a 5.6m steel truss using 11 Imote2s
  - Imote2 necessary due to high computational complexity.
  - It worked! [RTSS 2008]

Problem: Imote2 is power-hungry

Solution: Use middleware to exploit device heterogeneity by integrating energy-efficient nodes
Patient Monitoring

- 4-17% of hospitalized patients suffer clinical deterioration.
  - Wireless clinical monitoring system collect patients’ vital signs.
  - Difficult due to patient mobility.

- Dynamic Relay Association Protocol (DRAP) [SenSys 2010]
  - Automatically adapts to dynamic network topology
  - Real-time monitoring of mobile patients

- Implemented and underwent clinical trials at Barnes-Jewish Hospital in St. Louis.

- **Problem:** Application-specific
- **Solution:** Provide adaptation via middleware
Servilla

- An adaptive middleware for heterogeneous WSNs
- Introduces Service-Oriented Computing into WSNs

**Features**

- Multiple forms of service invocation
- Energy-aware service binding
- Service invocation sharing
- Adaptive binding
- Middleware asymmetry

**Notes**

- Device Heterogeneity
- Network Dynamics
- Limited Resources
Servilla’s Programming Model

**Application Scripts** (Platform-Independent & Interpreted)

- Medical Patient Monitoring
- Building Automation
- Security
- Structural Health Monitoring

**Services** (Platform-Specific & Native)

- Patient Sensors
- Occupancy Sensor
- Door/Window Sensor
- HVAC Actuators
- Energy Meter
- Vibration Sensor
- DSP

**Device Heterogeneity**

Dynamic discovery and binding using SOC

**Network Dynamics**

Limited Resources
Service-Oriented Computing (SOC)

Dynamic Service Discovery, Matching, and Binding
Evolution of SOC in WSNs

Atlas (King’06), TinySOA (Lopez’07), PhyNet (ArchRock’08), TinyWebServices (Pryantha’08), Servilla (Fok’09)

Enables In-Network Processing
Multiple Forms of Service Invocation

On-Demand

Periodic

Event-Based

These three types of service invocations are commonly used in WSN applications.

Selecting a suitable type of invocation can significantly increase energy efficiency.
Energy-aware Service Binding

- Selectively bind to the most efficient service provider
  - Possible due to device heterogeneity
- Requires additional information about energy efficiency
  - Provider sends 8 attributes to consumer during discovery
  - Consumer calculates and selects the most efficient provider

<table>
<thead>
<tr>
<th>Action / State</th>
<th>TelosB</th>
<th>Imote2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Idle (Sensor Off)</td>
<td>0.45</td>
<td>109.7</td>
</tr>
<tr>
<td>Idle (Sensor On)</td>
<td>--</td>
<td>204.83</td>
</tr>
<tr>
<td>Sensing</td>
<td>102.9</td>
<td>241.83</td>
</tr>
<tr>
<td>Message Tx</td>
<td>51.82</td>
<td>184</td>
</tr>
<tr>
<td>Message Rx</td>
<td>57.52</td>
<td>192.91</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Action / State</th>
<th>TelosB</th>
<th>Imote2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensing</td>
<td>18.49</td>
<td>2.61</td>
</tr>
<tr>
<td>Message Tx</td>
<td>725</td>
<td>506</td>
</tr>
<tr>
<td>Message Rx</td>
<td>6.17</td>
<td>14.07</td>
</tr>
</tbody>
</table>

Latency (ms)
Service Invocation Sharing

- Provider can optimize for energy efficiency by grouping multiple service invocations into one
- Reflected in the provider’s energy specification sent to the consumer
  - Process is opportunistic, automatic and hidden from consumer
Adap7ve Service Binding

- Middleware performs adaptive binding transparently from the application

Flushing service registry and re-discovering services after three consecutive failures
Lightweight Service Specifications

- Traditional specifications impose too much overhead
  - WSDLInterpreter (Vaughan’07) consumes 48KB memory
- ServillaSpec:

  Interface for Accuracy

  NAME = fft
  METHOD = fft-real
  INPUT = \{int dir, int numSamples, float[] data\}
  OUTPUT = float[]

  Attributes for Flexibility

  ATTRIBUTE Version = 5.0
  Attribute MaxSamples = 5000
  Attribute Power = 10

Implementation using TinyOS 2.x consumes only 1.4KB memory
Rapid Service Matching

- 10s of milliseconds to establish service match
  - Proportional to specification size

![Graph showing latency vs. specification size for Imote2 13MHz and TelosB](image-url)
Applications access services using a single line of code

- Hides the complexities of service discovery, binding, execution, and adaptation

1. uses AccelTrigger; // declare required service
2.
3. void main() {
4.   int flag = 0;
5.   while(flag == 0) {
6.     flag = invoke(AccelTrigger, “check”); // invoke service
7.   }
8.  // send alert…
9. }
Middleware Asymmetry

- Increases the types of devices supported
- Possible due to the decoupling of consumers and providers in the SOC programming model
- Hidden from the application developer

---

**Servilla - Virtual Machine Service Provisioning Framework**

- **Powerful platform**
  - CPU: 32-bit 416MHz
  - Memory: 32MB RAM
  - Power: 151mW

- **Weak platform**
  - CPU: 16-bit 8MHz
  - RAM: 10KB RAM
  - Power: 0.33mW

---

Washington University in St. Louis
Servilla’s Implementation

- On top of TinyOS 2.x
- Tested on Imote2 and TelosB devices

Scripts execute on VM to ensure platform independence

Service Provisioning Framework (SPF) modularized to enable middleware asymmetry
Memory Footprint on TelosB

- Nearly all of the memory on a TelosB is consumed by just the virtual machine!
- TelosB can still participate as a service provider
Structural Health Monitoring

- Do not continuously run DLAC due to high energy costs
- During idle periods use AccelTrigger
  - An energy efficient service
  - Periodically sense vibration to determine likelihood of damage
  - Only perform DLAC if potential damage exists
- Energy-aware service binding
  - Should AccelTrigger be bound locally or remotely?

- Local invocation
- Remote invocation
Significant Energy Savings Possible

- Energy-aware service binding is critical due to significant differences in energy efficiency among providers.
- In the structural health monitoring application, energy consumption can be reduced by up to 98.9%.
Validation of Energy-Awareness

- The consumer can determine the most efficient provider using the eight provider attributes
  - 10% radio duty cycle, 1Hz invocation and sensing frequencies
  - Bind to remote service when number of service executions ≥ 4

![Energy Footprint vs. Number of Service Executions](image-url)

- Local Invocation
- Remote Invocation
- Predicted vs. Actual Invocations
Medical Patient Monitoring

- Servilla hides the complexities of adapting to network topology changes due to patient mobility
- Using Servilla, the application implementation is trivial

```java
1. uses Sensor, DataRelay;
2.
3. void main() {
4.     while(true) {
5.         int data = invoke(Sensor, "get");
6.         invoke(DataRelay, "send", data);
7.         sleep(15); // sleep 15 seconds
8.     }
9. }
```
Experimental Setup

- WSN test bed containing 73 TelosB nodes deployed across two buildings at Washington University in St. Louis
- Patient travels along 359m path at two speeds
  - Slow: 0.7m/s average
  - Fast: 1.6m/s average
- Patient’s node invokes data relay service every 15s
Successful Adaptation

- Adaptive service binding enables the application to achieve 100% successful delivery of patient data
- Adaptation process hidden from the application developer
  - Abstracted by a simple invoke interface

<table>
<thead>
<tr>
<th>Service Invocation Success Rate</th>
<th>Servilla</th>
<th>CTP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fast Walk</td>
<td>100% ± 0%</td>
<td>31.6% ± 7.6%</td>
</tr>
<tr>
<td>Slow Walk</td>
<td>100% ± 0%</td>
<td>40.47% ± 11.2%</td>
</tr>
</tbody>
</table>
Conclusions

- Middleware can simplify sensor network applications while enhancing software flexibility and network efficiency

- Servilla supports service-oriented computing over sensors
  - Multiple forms of service invocation
  - Energy-aware service binding
  - Service invocation sharing
  - Adaptive binding
  - Middleware asymmetry

- Efficacy demonstrated via real-world applications
  - Medical patient monitoring – reliability despite mobility
  - Structural health monitoring – energy savings due to network heterogeneity
References
