Pervasive Computing and Communications

Mohan Kumar
The University of Texas at Arlington
kumar@cse.uta.edu
http://ranger.uta.edu/~kumar
http://cse.uta.edu/research/pico@uta

Students: H. Alex, G. Duffy, S. Kalasapur, M. Kim, B. Lagesse, N. Mallesh, D. Maxey, S. Patil, and K. Senthivel

This material is based upon work supported by the National Science Foundation under Grant Nos. STI-0129682 and IIS-0326505 and The Texas Higher Education Coordinating Board's Advanced Research Program

9/7/2010 Kumar
Organization

- Motivation
- Pervasive Computing
- Services in pervasive environments
  - Modeling services
  - Service Composition
- PICO project at CSE@UTA
- SeSCo – Seamless Service Composition
- Applications
- Conclusions
- Ongoing and Future work – Possible Collaborations
Mark Weiser’s prophecy

- “The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it.”

Scenario

M. Kumar et al.,
Pervasive Information Communities
Organization PICO: A Middleware Framework for Pervasive Computing,
Scenario

The scenario uses existing basic component technologies

Laptops, cameras, cell phones, PDAs etc.

The whole is much greater than the sum of its parts
How can pervasive computing help?

- Desired actions
  - Detect and recognize events
  - Recognize high-level event
  - Discover and deploy services
  - Combine services
  - Match services to resources
  - Address dynamic issues
  - ...

- On a **TIMELY, AUTOMATED, TRANSPARENT** basis

- Solution:
Organization

• Motivation

• **Pervasive Computing**
  • Services in pervasive environments
    – Modeling services
    – Service Composition
  • PICO project at CSE@UTA
  • SeSCo – **Seamless Service Composition**
  • Applications
  • Conclusions
  • Ongoing and Future work – Possible Collaborations
Pervasive Computing: Challenges

- **Proactivity and transparency**
  - Delays, resource utilization, unobtrusive services

- **Heterogeneity and interoperability**
  - Unevenness, incompatibility, h/w, s/w, communication channel

- **Location awareness and mobility**
  - Handoff- vertical/horizontal, data dissemination/acquisition

- **Authentication and security**
  - Privacy vs. services, cost, agents, active networks, availability

- **Smart environments**
  - Deployment, Interference

---

Enabling Technologies

- Networking Technologies
  - Registration and Auto-configuration
  - Mobility management
  - Active networking
  - Wireless access

- Computer Systems Technologies
  - Distributed systems
  - Mobile computing
  - Software agents
  - Information acquisition and dissemination

- Privacy, Trust, Authentication and Security

- Human-Computer Interfaces
  - Multi-modal
    - Voice, touch, GUI, brain-waves, implied command,
Middleware Services

- Glue heterogeneous entities
  - platform for interaction
- Match services to resources
  - Application specific
  - User profiles
- Combine resources and services
- Respond to user/application needs
- Mask unevenness
- Facilitate context-awareness
- Facilitate cooperation and collaboration
Organization

- Motivation
- Pervasive Computing

- **Services in pervasive environments**
  - Modeling services
  - Service Composition

- PICO project at CSE@UTA
- SeSCo – Seamless Service Composition
- Applications
- Conclusions
- Ongoing and Future work – Possible Collaborations
Service Composition

- Combining basic services into possibly complex services
- Typically works on the template matching principle
  - Requirements are specified in the form of a template
  - Runtime environment locates services that fit the place holders in the template
  - Coordination among identified services is performed by the runtime environment.

- Complex tasks can be broken down into subtasks
  - One template for each subtask
- Service discovery mechanisms available to locate available services
- Invocation of identified services
  - Event based
  - Process based
  - Task dependant
Service composition in pervasive computing

• Two types of composition mechanisms
  • Static
    – Composition orchestrated prior to need
    – Capability to define finer interface dependency details
    – Ideal for stable, managed environments
    – Insufficient support for dynamism
  • Dynamic
    – Composition formed once the request arises
    – Can consider the current context/service availability
    – Costly in terms of time for composition.

Current Work:
Research initiatives, no commercial implementations

   Made to support systems, limited to pre-designed solutions

Service composition used as a delivery vehicle for applications.
Majority work on discover + match style
Service composition in pervasive computing: Discover + Match

- Identify suitable services, mediate interactions, transactions among identified services
  - Spidernet\(^1\), Reactive composition\(^2\), Konark\(^3\)
- Mostly variations of template matching schemes


Organization

- Motivation
- Pervasive Computing
- Services in pervasive environments
  - Modeling services
  - Service Composition
  - Hierarchical Service Composition

- **PICO project at CSE@UTA**
  - SeSCo – Seamless Service Composition
  - Applications
  - Conclusions
  - Ongoing and Future work – Possible Collaborations
PICO Project at CSE@UTA

• Pervasive Information Community Organization
  – Provides transparent, automated services: *what you want, when you want, where you want, and how you want.*

• PICO is a framework to create *mission-oriented* dynamic computing communities of software agents that perform tasks on *behalf of users* and devices *autonomously* over existing heterogeneous network infrastructures, including the Internet.

• Propose concept of “COMMUNITY COMPUTING” to provide *continual, dynamic, automated and transparent* services to users.
PICO Architecture

Telem medicine

Manufacturing

Smart home

Community

Delegents

PICO Middleware Services

Devices

Access point/Gateway

Bluetooth 802.11b Cellular

…

Access point/Gateway
Basic Building Blocks of PICO

- **Physical devices** –
  - Computer-enabled devices: small, wearable to large supercomputer
  - Sensing capabilities
  - Computational power
  - Communication capability
  - Actuators

- **Software entities** – **Delegents (Intelligent Delegate)**
  - Intelligent SW agents – service provisioning
  - Proxy-capable: exist on the infrastructure
  - Event-driven
  - Execute on host devices
  - Need a host for execution
  - Mobile, capable of communicating
Community Computing: PICO based middleware

- **Formation of Computing Communities**
  - Community – Set of collaborating delegents
  - Service provisioning
    - Programmed/scheduled
    - Continuous services
  - Dynamic
    - Membership and community defined a priori
    - Goal-driven
    - Dynamically formed
    - Dynamically dismantled
  - Dynamically formed based on events
Devices

- $C = < C_{ID}, S, F >$
  - $C_{ID}$: Device identifier
  - $S$: System characteristics
  - $F_c$: Functionality of device
- For example, $C = $ Heart Monitor
  - $S = < \text{operating system}; \text{processor type}; \text{memory}; \text{I/O type}; \text{battery}; \text{wireless transceiver} >$
  - $F = < \text{ECG monitoring}; \text{processing}; \text{communicating}>$
Delegents

• Intelligent Delegate: works diligently on behalf of a camileun, user, application or service

• $D = < D_{id}, F_{d} >$
  
  – $D_{id}$: Delegent ID: $< Id, C, P >$
    • $Id$: Delegent ID
    • $C$: Host Device ID
    • $P$: Community
  
  – $F_{d}$: Functionality of delegent: $< M, R, S >$
    • $M$: Program modules
    • $R$: Delegent rules
    • $S$: Delegent services
Delegent Example

• A Delegent for ECG monitoring
  – Functionality: <Modules, Rules, Goals>
  – **Modules**: Signal processing module, Arrhythmia detector, Software filter, Timer, Communication module.
  – **Services**: Detect arrhythmia, Upload ECG window, Communicate status.
Community

- A community is defined by $P = \langle U, G, E \rangle$
  - $U$: set of delegents in the community.
  - $G$: community goal(s).
  - $E$: community characteristics.
    - Community ID
    - Number of delegents
    - Community coordinator/manager
      - Joining/disjoining of delegents
      - Location awareness and other services
    - Community of delegents need resources
      - CPU, Memory, Bandwidth

CPU, Memory, I/O
Devices + Delegates = Services on Resources

- Surveillance
- Network Node
- Information Kiosk

Network service community
- Visitor’s delegate

Police community
PICO : Recap

• A middleware framework aimed at providing transparent, automated support to users
  – Creates a transparent platform over heterogeneous devices.
  – Device features and specific software functionalities exported as services
  – Creation of communities of software agents to support mission critical tasks.
  – Event oriented operation

• Three basic constructs
  – Devices (C = <D_id, H, F>)
  – Delegents (D = <D_id, M, R, S>)
  – Communities (P = <P_id, U, G, E>)
Organization

• Motivation
• Pervasive Computing
• PICO project at CSE@UTA
• Services in pervasive environments
  – Modeling services
  – Service Composition
  – Hierarchical Service Composition

• **SeSCo – Seamless Service Composition**
• Ongoing and Future work – Possible Collaborations

Service model

• Each service is treated as a transformational unit, accepting a set of inputs and producing a set of outputs
  – Derived from the state machine representing delegent operation
• Each service faithfully works towards its desired goal
  – Assumption -- there are no malicious services
  – Security and trust schemes need to be employed to enhance the model
• Services are represented using a directed, attributed graph
• Each service has a set of attributes associated
  – Each service is described using both semantic and syntactic attributes
Service model

- Each service is represented by
  \[ G_S = \{V_s, E_s, \mu_s, \xi_s\} \]
  \( V_s \rightarrow \) node(s) representing service (or the state machine for the service)
  \( E_s \rightarrow \) Edges to and from the service (I/O).
  \( \mu_s \rightarrow \) Vertex attribute function.
  - Service Name, location, address, cost, etc.
  \( \xi_s \rightarrow \) Edge attribute function
  - Type and size of parameters and messages
- Attributes can also contain semantic descriptions of entities.

Stype: Text
PType: ASCII
Rate: 56kbps

Text to audio
Location: 238 ELB
Cost: $0.5/min
Delay: 0.05mS

Stype: Audio
PType: Wav
Rate: 56kbps
Service aggregation at the directory

• Aggregation based on the transformation achieved by each node
• Stored as a two-layered directed, attributed graph
  – First layer stores semantic transformations
  – Second layer stores syntactic transformations
• At the second layer, each available service is represented using a directed edge between the parameters it transforms
Service aggregation at the directory

Service graphs

Semantic aggregation

Syntactic aggregation

Aggregation of services \((G_p)\)

9/7/2010 Kumar
Task specification

• Each task/request represented as \( G_R = \{V_r, E_r, \mu_r, \xi_r\} \)
  
  \( V_r \rightarrow \) Vertex set representing individual services required
  
  \( E_r \rightarrow \) directed edge set representing interactions among the requested services
  
  \( \mu_r \rightarrow \) vertex attribute function
    – Service name, allowable cost, required bandwidth, permissible delay, etc,
  
  \( \xi_r \rightarrow \) Edge attribute function
    – Details about exchanged messages such as size, expected format, data rate, etc.

Stype: document
  PType: ASCII, pdf, doc
  Size: 4Mb

Stype: Text
  PType: ASCII
  Rate: 56kbps
  Location: ELB
  Cost: $0.7/min
  Delay: 0.1mS

Stype: Audio
  PType: Wav
  Rate: 56kbps
  Cost: $0.05/min
  Delay: 0.05mS
Task resolution

- Each request \((GR)\) consists of a number of services that need to be identified
- Consider each node, and its I/O types
  - Semantic and syntactic representations
- Look for a path in the semantic aggregation
  - If a path exists, composition possible
  - Locate the shortest path (k shortest paths)
- Locate corresponding paths in the syntactic aggregation
  - Weights defined by user preferences, quality parameters, cost, etc., as attributes of \(G_R\)
- Perform the procedure for all nodes in \(G_R\)
Task resolution

Stype: A  Ptype: a1
Stype: B  Ptype: b
Stype: T  Ptype: t1
Stype: M  Ptype: m
Stype: K  Ptype: k
Stype: T  Ptype: t
Service aggregation process
(Centralized scheme, managed networks)

- Well known directory structure
  - Services register at the directory
  - Directory maintains all available services by aggregating them
- Directory acts as the point of contact for task support
- Service composition performed at the directory
  - Execution of composition left to the client
PICO Abstractions

\[ P = \langle U, G, E \rangle \]
\[ U = \{\text{voice capture } \rightarrow (c), \text{ Alert, Call contact, file system}\} \]
\[ G = \{\text{auto phone, file browse, file save}\} \]
\[ E = \{\text{phonebook, 30sec data...}\} \]

\[ \forall u_i \in U, \exists s_j \in S(D) | u_i \approx s_j \]

\[ D = \langle M, R, S \rangle \]
\[ M_{hs} = \{\text{sample voice, send voice, render voice} \rightarrow (b)\} \]
\[ R_{HM} = \{E = "\text{sample voice}"; C = "\text{sample ready}"; A = "\text{render voice}"\} \]
\[ S = \{\text{audioOut, voice capture}\} \rightarrow (a), (c) \]

\[ D = \langle M, R, S \rangle \]
\[ \forall m_i \in M(D), \exists h_i \in H(C) | \text{req}(m_i) \approx i \]
\[ \exists s_i \in S(D) | s_i."\text{project}f_j \in F(C) \]

\[ C = \langle H, F \rangle \]
\[ \forall d_i."\text{installed n'}C_j, \exists H(C) | M(d_i) \approx \{H\} \]

\[ C = \langle H, F \rangle \]
\[ C_{\text{Headset}} = \langle H_{\text{Headset}}, F_{\text{Headset}} \rangle \]
\[ H_{\text{HM}} \{\text{mem = 3MB, power_level = normal, communication = bluetooth } \rightarrow (b)\} \]
\[ F_{\text{HM}} \{\text{audible voice, voice capture}\} \]
Service Layer

Manages all communication with the service layer

Service Aggregation
- Performs aggregation of discovered services along with local services
- Graph aggregation

Service store
- Stores aggregation of services in a layered graph

Service Composer
- Performs composition of request based on aggregation.

Incoming
- Ads
- Advertise
- Request
- Result

Service Manager
Service aggregation process
(Centralized scheme, managed networks)

- Well known directory structure
  - Services register at the directory
  - Directory maintains all available services by aggregating them
- Directory acts as the point of contact for task support
- Service composition performed at the directory
  - Execution of composition left to the client
Centralized scheme: *limitations*

- Ideal to build and operate Service Provider communities, managed infrastructures
- Directory structure can be distributed, but all requests need to be resolved at the directory
- Ideal to build SOAs around assets in the infrastructure
  - Managed service definitions
  - Managed resources, therefore managed directory
  - Access to different resources can be controlled
- Directory node becomes the bottleneck
  - No explicit support for heterogeneous resources
  - No explicit support for dynamisms
Seamless service composition (SeSCo)

• Main Motivation
  – Resource heterogeneity to be actively exploited
  – Resource poor devices need to be proactively supported by their resource rich counterparts

• User centric solutions
  – ‘What, where, and how’ type of services
  – Dynamisms associated with user mobility
  – Quality of composed service
## Device classification chart

<table>
<thead>
<tr>
<th>Level (α)</th>
<th>Middleware version</th>
<th>Features</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>None</td>
<td>Features exported through delegates on Level-2 or Level-3 devices, No native personalization support</td>
<td>sensors, Legacy printers</td>
</tr>
<tr>
<td>1</td>
<td>Minimum</td>
<td>Community member, can not be a proxy, possibly mobile</td>
<td>Cell phone, mote sensors, smart printer</td>
</tr>
<tr>
<td>2</td>
<td>Complete</td>
<td>Community member, leader, Can act as a proxy, possibly mobile, Rsource rich</td>
<td>Laptop, PDA</td>
</tr>
<tr>
<td>3</td>
<td>Complete</td>
<td>Community member, leader, Can act as a proxy, Rsource rich, typically resides in the infrastructure, managed</td>
<td>PC, Server, Grid, cluster</td>
</tr>
</tbody>
</table>
Creating the hierarchy

The LATCH protocol

- Devices lower in the hierarchy *latch* onto those higher in the hierarchy
- Service related information (Service graphs) exchanged during hierarchy creation
- Staged aggregation for improved composition support.
- Status changes reflected through periodic updates.
Hierarchical service aggregation

- Devices at each level in the hierarchy *aggregate* all services from that part of the hierarchy
  - Includes services available on the local device
  - Plus the services available at all descendent nodes
Properties of the overlay

- **Service zone**
  - All the services that a device is responsible for

- **Search zone**
  - All devices where a possible match is searched for

Requests resolved by propagation through the hierarchy

- Search zone is *expanded* incrementally
- Partial composition is computed at each expansion
- Unresolved parts of the requests are forwarded to the parent node

Compositions in the *local service zone* handled by the local node.
Hierarchical service composition
Hierarchical service composition

Request at D3
Comparison with D + M approach

Graph based Composition
Discover + Match

% Composition success vs. # Available services

Competitive ratio vs. # Available services

0 20 40 60 80 100 120
0 1 2 3 4 5
50 100 150 200 250 300 350 400 450 500 550 600 625

50 100 150 200 250 300 350 400 450 500 550 600 625

9/7/2010 Kumar
Service composition success rate
Based on the composition length

% Composition success

Composition length 2
Composition length 4
Composition length 5
Composition length 6

# Registered services
User Mobility

- User mobility is inherent to pervasive computing environments
- When a user moves in the middle of a service session, continued service support needs to be ensured
  - By reconstructing the part of the service that was disrupted
  - Usually rendering services need to be reconfigured
Resource mobility

- Mobile resources
- Resource limitations
  - Battery power, increased load, memory shortage

- Parts of the composition which are no longer valid need to be recomputed
  - The disconnected node could be a higher level node
  - More taxing as compared to user mobility

![Graph showing the comparison between Hierarchy based and Broadcast based methods](image-url)
SOA: An Example

Techno-rich automobile, services representing device features

9/7/2010 Kumar
SOAs in action

- Tasks can be executed using available services
- Tasks represented as directed, attributed graphs
  - Nodes represent required services
  - Edges represent transformations among services
- Task graph acts as a template for locating and utilizing services

Example tasks

1. E-mail message
   - Text
   - Display

2. URL
   - HTML
   - Display

3. AV File
   - Audio stream
   - Sound
Comparing delivery mechanisms

E-mail message → Email service → Text → Laptop display

Composing with D+M

E-mail message → Email service → Text → Text to image → Display

Composing with SeSCo

E-mail message → Email service → Text → Text to image → JPG → Video display

E-mail message → Email service → Text → Text to image → JPG → GPS display
Modeling devices within an automobile

- Identify available devices
- Create a network among devices
- Create services representing identified features
- Identify features

- Video capture
- Audio
- Display
- Calendar scheduler
- Text to voice
- Communication
- Audio capture
Prototype details
Available and accessible services

9/7/2010 Kumar
PICO Middleware

Delegent specs

Device's Delegent Manager

Collaboration Manager

Event Handler

Resource Manager

Service Layer

Adaptation (Library)

Device specs

TCP/IP Bluetooth IR USB
Organization

- Motivation
- Pervasive Computing
- PICO project at CSE@UTA
- Services in pervasive environments
  - Modeling services
  - Service Composition
  - Hierarchical Service Composition
- SeSCo – Seamless Service Composition

- Conclusions
- Ongoing and Future work – Possible Collaborations
Conclusions

• SOAs to build pervasive computing systems
  – Resources exported as services
  – Service composition mechanism to deliver task support

• SeSCo capable of providing advanced support
  – Capability to dynamically weave composite services from basic ones
  – Maintains locality of service, quality of composition
  – Capable of handling user and resource mobility

• Hierarchical service overlay
  – Formed based on resource states
  – Enables resource poor devices to exploit presence of powerful devices

• Dynamic mechanism to maintain overlay and composite services
  – Reflects current resource status
  – Can maintain composite service sessions
Publications

- S. Kalasapur, M. Kumar, B. Shirazi, “Evaluating Service Oriented Architectures in Pervasive Computing,” In proceedings of the fourth international conference on Pervasive Computing and Communications, PerCom 2006, Pisa, Italy.
- S. Kalasapur, M. Kumar, B. Shirazi, “Seamless Service Composition (SeSCo) in Pervasive Environments.” In the proceedings of the first international workshop on Multimedia Service Composition, MSC 2005, November 2005, Singapore.
- S. Kalasapur, M. Kumar, B. Shirazi, “Personalized service composition for Ubiquitous Multimedia Delivery.” In the proceedings of the international conference on the World of Wireless Multimedia, WoWMoM 2005, Taormina, Italy.