A Place-aware Spontaneous Service Provision Framework for IoT Enriched Smart Spaces

- Internet of Things (IoT): Beyond Connectivity Workshop -

Prof. Dongman Lee
Department of Computer Science in KAIST
dlee@cs.kaist.ac.kr

May 29th, 2014
1. Motivation
   - Paradigm shift from a single “Place” to “Places”
   - Placeness: beyond the intrinsic functionality of a place

2. Related Works

3. Place-aware Spontaneous Service Provision
   - Placeness-based Task Discovery
   - QoS-aware Task Selection

4. Implementation
   - IoT enriched testbeds in public places

5. Evaluation
   - Task recommendation accuracy
   - Response time for task execution

6. Conclusion
Everyday consumer electronics become smart objects
  - e.g.) Smartphones, robot cleaners, smart TVs, refrigerators, air conditioner, etc.

Hence, ubiquitous computing environments are extended
  - From a single well-known place to multiple unknown public places

This extension causes the following problem

- **High complexity** on identifying appropriate services on the fly
  - Because of increasing possible sets of service composites via various smart objects
  - As well as a place can have several meanings as people conduct social activities involving smart objects in the place over the time ("Placeness")
**Why “Place-aware Service Provision”?**

- Traditional spontaneous service recommendation and provision
  - Many research efforts to revolve the complexity problem
  - e.g.) Kawsar et al. [2, 3], TaskOS [4], etc.

- Provide customized services based on pre-defined rules and usage history in a given place
- However, it is unreasonable to **predefine all the possible user tasks** that may occur in the place

> Many of **unforeseen opportunities** to be served with IoT may be lost
Therefore, we propose …

- A place-aware spontaneous service provision framework
  - First phase: Placeness-aware task discovery
    - Infers possible tasks people have conducted in the given place
    - By matching the user’s social characteristics against those of the people
    - Further finds the tasks from other places whose types are the same as the current place
  - Second phase: QoS-aware task selection
    - Evaluate the task candidates
      - Since the configuration of the current place may not be the same as similar places
    - Measure how closely the smart objects in the current place can meet QoS constraints specified as each user’s preferences
      - e.g.) SLA (Service Level Agreement)
    - Selects only feasible tasks and prioritizes them according to their qualification

- We tested the proposed scheme on our three testbeds
  - Smart objects are equipped with beagle board PCs and a place server exist
  - Experiment results show that the proposed scheme finds appropriate tasks in the testbeds in a reasonable time
Spontaneous Service Recommendation and Provision

- Qin et al. [6]
  - Introduce the term ‘spontaneous service provision’
  - Propose impromptu service discovery (a single service)
  - Limitation: most user tasks are too complex to be supported via a single service

- Dong et al. [7]
  - A well-designed architecture for spontaneous interaction with smart objects
  - Limitation: only identify requirements for spontaneous service provision without implementation

- Kawsar et al. [2, 3]
  - Propose spontaneous federation of smart objects using structural type matching to associate services and smart objects
  - Limitation: conditional rules should be defined prior to recommendation

- TaskOS [4]
  - Context-aware task recommendation based on collaborative filtering technique
  - Limitation: spontaneous services available in a given place but never have been performed
Spontaneous Service Recommendation and Provision

- Qin et al. [6]
  - Introduce the term ‘spontaneous service provision’
  - Propose impromptu service discovery (a single service)
  - Limitation: most user tasks are too complex to be supported via a single service

- Dong et al. [7]
  - A well-designed architecture for spontaneous interaction with smart objects
  - Limitation: only identify requirements for spontaneous service provision without implementation

- Kawsar et al. [2, 3]
  - Propose spontaneous federation of smart objects using structural type matching to associate services and smart objects
  - Limitation: conditional rules should be defined prior to recommendation

- TaskOS [4]
  - Context-aware task recommendation based on collaborative filtering technique
  - Limitation: spontaneous services available in a given place but never have been performed

Limitations on previous works
- Do not concern about enhancing service provision opportunity for the current place
- Do not concern availability of the task according to currently available resources of smart objects in the current place
**Overall architecture**
- Entities: a user’s Smartphone, Smart objects, a Place server

**Main components**
- **Smart Object Manager**: collects service lists that each smart object supports and their status via Service Interaction Broker
- **Knowledge Base**: includes semantic sources from Linked Data, the status and usage statistics of smart objects (sensors and actuators), and the user’s type
- **Context Manager**: allows Placeness Miner to query users’ visiting histories and experiences
- **Task Manager**: uncovers potential tasks and prioritize them according to the QoS qualification when a user visits the place

*Figure 1. Overall architecture of the proposed scheme*
In-situ knowledge base construction

- Place semantics: what facilities and functionalities the place can provide
- Place server and smartphone share the place ontology
- Whenever the user visits a place and uses smart objects, Knowledge Base collects the data

**Phase 1: Placeness-aware Task Discovery (1/2)**

**Internet of Things (IoT): Beyond Connectivity**

![Diagram of a place ontology and knowledge construction](image)

*Figure 2. Place ontology and knowledge construction*
Phase 1: Placeness–aware Task Discovery (2/2)

- Placeness-based task finding
  - To discover tasks that the user can conduct in the given place
  - Leverage a collaborative filtering method based on similarity of user contexts

- Find either in-situ tasks being fulfilled by nearby similar users or conventional tasks performed by visited users

- Expand the placeness into similar one in other places → tasks in the similar place becomes candidate tasks for recommendation

- Rank tasks by measuring the relevance of user experiences
  - what type of users have been to a place
  - what social activities have frequently been done
Phase 1: Placeness-aware Task Discovery (2/2)

- Placeness-based task finding
  - To discover tasks that the user can conduct in the given place
  - Leverage a collaborative filtering method based on similarity of user contexts

\[ \text{UserSimilarity}(U_i, U_j) = \frac{|UDS_i \cap UDS_j|}{|UDS_i \cup UDS_j|} \]

\[ \text{PlaceSimilarity}_{ij} = \cos(\text{SMO}_i, \text{SMO}_j) + \frac{|PDS_i \cap PDS_j|}{|PDS_i \cup PDS_j|} + \cos(\text{FPC}_i, \text{FPC}_j) \]

\[ \text{TaskSimiliarity}_{ij} = \frac{\text{TSA}_i \cap \text{TSA}_j}{|\text{TSA}_i \cup \text{TSA}_j|} + \sum_{n=1}^{N} w_n \times \cos(\text{sos}_i, \text{sos}_j) \]

\[ \text{TaskScore}_{ij}^k = \alpha \times \text{TaskPopularity}_{ij}^k + (1 - \alpha) \times \text{InSituPlaceness}_{ij}^k \]
After finding possible tasks in the given place...
- The proposed scheme measures how closely the smart objects can meet users’ preferences specified in a form of SLA
  - e.g.) SLA for “Watching a Movie” task

Therefore, the proposed scheme resolves this problem dividing a global optimization problem as multiple local optimizations
- First, the proposed scheme divides the quality range into multiple sub-regions
- Then, the proposed scheme finds a representative service with the highest utility value
- Lastly, the proposed scheme filters out tasks whose aggregated QoS doesn’t satisfy given QoS constraints

However, finding an optimal set of services for each task can be regarded as a global optimization problem (known to be an NP hard problem [11])
Dividing the given quality range

- Calculates average quality values of each abstract service
- Distributes constraints into abstract services according to their average ratios
  - Each distributed constraint is called ‘estimated quality’

\[
\text{Average Quality Value}^h_k = \frac{\sum_{i=1}^{l} a_{ik}}{l} \quad (1)
\]

\[
\text{Estimated Quality}^h_k = \frac{AQV^h_k}{\sum_{k=1}^{m} AQV^h_k} \cdot \text{Global Constraint}^h_k \quad (2)
\]

where \((1 \leq k \leq m), (1 \leq h \leq n)\)

- Equation (1) takes the average value of a quality attribute in an abstract service
- Equation (2) estimates the ‘plausible’ local constraints of quality attributes for each abstract service

Divides quality ranges with different intervals

- If constraints become tighter
  - Estimated quality are assigned at high quality
  - Then, the quality range is divided more densely in high quality

---

**Internet of Things (IoT): Beyond Connectivity**
Finding optimal service composite
- Chooses the representative service with the highest utility value

\[
U(s_i) = \sum_{k=1}^{r} \frac{q_{\max}(k) - q_k(s_i)}{q_{\max}(k) - q_{\min}(k)} \times w_k
\]  

(3)

\[q_{\max} \text{ : Max. of the aggregated QoS attribute values}
\]
\[q_{\min} \text{ : Min. of the aggregated QoS attribute values}
\]
\[q_k \text{ : } k^{th} \text{ quality attribute value}
\]

- Finds the optimal set by inspecting all combinations of the best service candidates (Integer Programming [11])

\[
\text{maximize } \sum_{j=1}^{n} \sum_{i=1}^{l} U(s_i) \times x_{s_i}
\]  

(4)

\[n \text{ : The number of abstract services}
\]
\[l \text{ : The number of sample services}
\]

Selecting feasible tasks
- Aggregates QoS of the optimal set of services for each task
- Then drops tasks whose aggregated QoS does not satisfy the given QoS constraints
- Finally, prioritizes the rest of the tasks by comparing the QoS qualification

\[
\forall k: \sum_{j=1}^{n} \sum_{i=1}^{l} q_{s_i,k} \times x_{s_i} \leq C_k
\]

(5)

\[n \text{ : The number of abstract services}
\]
\[q \text{ : } k^{th} \text{ quality value of service } i \text{ at } j^{th} \text{ abstract service}
\]
\[C_k \text{ : } k^{th} \text{ global constraints}
\]
**Construction of the Testbeds (1/3)**

- Two different smart seminar rooms and one smart lounge
  - Each of which is equipped with a different set of smart objects
  - Existing objects were empowered with Beagle board PCs running Ubuntu 11.10
  - Various sensors for tracking physical status of each object

<table>
<thead>
<tr>
<th>Testbed</th>
<th>Place Characteristics</th>
<th>Group Characteristics</th>
<th>Available Smart Things</th>
<th>Performed Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Seminar Room CS1408; room; students; education; Daejeon;</td>
<td>Occupation: Student, Age: 18<del>28, # of users: 3</del>7, Common Interests: Samsung Camera, Exercise, Running</td>
<td>TV, Curtain, Projector, Screen, Air Conditioner, Printer, Curtain, Web Cam</td>
<td>1. Having a lab meeting 2. Having discussion 3. Watching movies 4. Eating food 5. Sharing materials 6. Having a video conference</td>
</tr>
<tr>
<td></td>
<td>Seminar Room CS825; room; students; education; meeting; Daejeon;</td>
<td>Occupation: Student, Age: 24<del>28, # of users: 6</del>10, Common Interests: Research, Game, Running, Playing</td>
<td>Projector, Screen, Air Conditioner, Web Cam, Flower Pot, Vacuum Cleaner</td>
<td>1. Having a lab meeting 2. Having discussion 3. Studying</td>
</tr>
<tr>
<td></td>
<td>Rounge N802; room; students; relaxation; chattering; Daejeon;</td>
<td>Occupation: Student, Age: 24<del>50, # of users: 1</del>5, Common Interests: Research, Game, Running, Playing</td>
<td>Vending machine, TV, Refrigerator, Air Conditioner, Web Cam</td>
<td>1. Having discussion 2. Watching movies 3. Eating food 4. Playing games 5. Listening to music</td>
</tr>
</tbody>
</table>

---

Internet of Things (IoT): Beyond Connectivity
Smart object discovery

- JmDNS – discover smart objects in the same network [13]
- The user’s smartphone connects to a local network formed by WAP (Wireless Access Point) and receive multicast advertisements from smart objects

Communication between smart objects

1. ODH (Object Description Handler) adds on URI of smart object to the DNS service discovery message as an attribute of services available at a given smart object
2. ODH provides dynamic service description by updating contextual values such as availability of each service
3. The smartphone constructs a service registry for executing a service described in a service execution sequence

Figure 4. Distributed Object Discovery Module
A simple application – “Task Recommender”

- Provides a user interface for showing discovered tasks in a given place
  
(a) When a user selects a task in the list, the client shows an execution sequence of the task

(b) The user executes each service in sequence instead of executing the task as a whole

(c) For example, a presentation begins by turning on a projector, sending a presentation file to the project, and starting a presentation control application on the user smartphone when the user selects the “Having a Presentation” task

Figure 5. Task Recommender: (a) a list of recommended tasks, (b) an overview of ‘Having a lab meeting’ task, (c) an individual controller for a projector
Demonstration Scenario

- **Situation: when a user visit CS825**
  - Similar placeness between CS1408 and CS825
    - Place characteristics: seminar room, education, Daejeon
    - Group characteristics: student, the twenties
    - Available smart objects: projector, air conditioner, web cam
  - The proposed scheme incorporates user experiences at CS1408 as implicit possible tasks
    - e.g.) “Watching a Movie”, “Having a Video Conference”, “Having a Presentation” tasks, and etc.
  - The proposed scheme checks the availability of each tasks and disregards “Watching a Movie” task

\[
Q_{\text{Projector,CS825}} = \{\text{bandwidth: 10Mbps, latency: 700ms, resolution: 1920x1080, brightness: 30lux}\}
\]
\[
C_{\text{movie}} = \{\text{bandwidth: 15Mbps, latency: 1000ms, resolution: 1920x1080, brightness: 60lux}\}
\]

\[
\sum_{i=1}^{4} Q_{i,\text{CS825}}^4 \cdot w_i^4 \leq C_{\text{movie}}
\]

(We assume that weights for each quality attribute are same)

- As the result, the proposed scheme selects the rest of tasks such as “Having a Video Conference” task and “Having a Presentation” task for the user
The effectiveness of placeness-based task finding

\[ TaskScore_{ij}^k = \alpha \times TaskPopularity_{ij}^k + (1 - \alpha) \times InSituPlaceness_{ij}^k \]

- Compare with baselines with TaskPopularity + InSituPlaceness ($\alpha = 0$)
- As shown in Figure 6, the proposed method is more accurate than others
- In-situ contextual information is more meaningful to infer the user’s task intention

The optimal value of $\alpha$

- As shown in Figure 7, when $\alpha$ is 0.8, the accuracy is highest
- In-situ situational place semantics is more important than diachronic place functionality
Criteria to measure efficiency on finding possible tasks
- Time overhead for service recommendation
- Time overhead on communication among smart objects

Experiment environment
- A user smartphone (Android OS 2.2)
- Four possible tasks in the testbed (CS1408)

All the tasks found within a reasonable time
- Within a second
- Including transmission time to get the information from smart objects

As the number of required smart objects increases, so does the service recommendation time

Figure 8. Performance on the testbed
Main contributions

- Proposed an opportunistic service recommendation scheme in a given place, which is organized with two phases
  - 1. Task discovery: the proposed scheme firstly discovers possible tasks ...
  - 2. Task selection: the proposed scheme then prioritizes the tasks according to their QoS qualification and recommends highly ranked tasks

- Demonstrate the feasibility of the proposed scheme by performing an example scenario on three different testbeds

Future works

- Carry out intensive user studies to measure the user satisfaction on the recommendation results
  - Build up more smart objects with various service scenarios
- Aid personalization feature to the current recommendation scheme
  - Users may have different preferences on various quality attributes
- Ultimately, develop the context-aware opportunistic task recommendation scheme to provide users with personalized services
감사합니다
References (1/2)


