Ontology Based Context-Aware Service Discovery for Pervasive Environments

Pravin Pawar
ASNA Group
Department of EEMCS
University of Twente
The Netherlands

Andrew Tokmakoff
Telematica Instituut
The Netherlands

Presentation during SIPE 2006, Lyon, France
29 June 2006
Outline

• Introduction to Service Oriented Architecture
• Service discovery
• Motivation
• Requirements for Context Aware Service Discovery (CASD)
• Architectural design of CASD
• CASD working
• CASD context sources
• Technical realization
• Related work
• Conclusion and future work
Service Oriented Architecture

- Collection of interacting services to achieve a common goal
Service Discovery

- A service is syntactically well-defined functional behaviour offered by a software entity for use by other software entities.
- The service advertises its service description for potential clients.
- A client interested to access the service obtains information about the existence of a service, its applicable parameters and terms through service discovery.
- The service registry facilitates the interaction between service and client.
- Due to technological advances mobile devices are also capable of utilizing services.
- However, mobile application needs to deal with selecting the best service considering its requirements.
Motivation

• Current service discovery protocols do not consider the context of services and clients during the service selection process!
• The work in (Broens T. et. al. 2004) shows that the contextual information makes the user’s query more information-rich and increases the precision of the retrieved results
• The discovery service may make use of context information to optimize the discovery process in two ways:
  – To search the most relevant services based on service context and client context
  – Monitoring the service and client context changes to suggest better services to the client
Requirements for Context Awareness
Service Discovery (CASD)

• Context information is highly interrelated and has many alternative representations
  – Context information should be represented unambiguously
• Services and clients in the pervasive environment may have multiple context sources in heterogeneous networks
  – Need for standardized support for context distribution
  – The mobile devices should be able to disseminate context information seamlessly in the pervasive environment
• Context information is dynamic (subject to change)
  – Support for notifications whenever context changes
  – The client should be transparent to the change in service context change
Our Contribution to CASD

• Once-off service discovery
  – Determine the most suitable service by taking into account the context information of both, the service and client
• Persistent service discovery: Service discovery is re-triggered when
  – When context of the service or client changes
  – In the event of the appearance of new services
• Standardized support for context distribution and notification
  – Modeling context sources as services
  – Context source notifies change in the context information
• Mobile context sources
  – Context source on a mobile device can participate in the fixed network as a nomadic mobile context source
• Unambiguous representation of context information
  – Use of ontologies
Architectural Design of CASD
Once-off CA Service Discovery

1. Query service directory for client’s desired service type
2. Obtain the context information of the services and store it in the service graph
3. Obtain client context information and append it to the service graph
4. Query service graph using client query
5. Return the most suitable service matching the client criterion (if found)
Persistent CA Service Discovery

1. **Query service directory for client’s desired service type**
2. **Obtain the context information of the services and Store it in the service graph**
3. **Obtain client context information**
   - **And append this to service graph**
4. **Store service graph in DB**
5. **Query service graph using Client query**
6. **Return the desired service which Match the client criterion (if found)**
7. **Send service change event**
   - If new service is different Than the previous one
8. **Context change Event received**
9. **Modify service Graph**
   - Using latest context info.
10. **Retrieve service Graph from DB**
11. **Subscribe service and Services context change event**
12. **Subscribe client context change event**
13. **Wait for Context change Event**
Representation of Context Using Ontology

- Ontologies are published using OWL
- A resource in OWL is represented as a class, and the relationship between resources is shown using properties.
- To represent context information, a context source must create an individual of the class and connect two individuals by the property.
Context Sources

Service directory

1. CS registration
2. Reference
3. CS reference
4. CS reference
5. CS reference
6. CS information
7. Context change subscription
8. Context change notification
9. Get context

Context source

Client or Service

CASD service

Context Information as XML tree
Technical Realization ...

- OWL Ontologies
- Client
- CASD Service
- Jini Lookup Service
- Jena RDF Graph Operations APIs
- MySQL Database
- Services
- Mobile Context Sources
- Fixed Context Sources
Nomadic Mobile Context Source

- Mobile Service Platform extends SOA to mobile devices so that they are able to host the service (Presented during WiMob 2006 in Montreal)
Related Work

• CASD is an active research area
• It promises to offload the client from the responsibility of selecting the appropriate service from a potentially large number of candidate services
• [Lee C. et. al.] propose to use context attribute as a flexible means to exploit relevant context information during the service discovery process.
  – We consider client context information using the client's context source instead of relying on static Service Registry context information.
• Reactive discovery proposed by [Capra L. et. al.] also triggers re-evaluation of services based on the context changes.
  – However, in our case, the client does not have the responsibility of re-evaluation
Conclusion and Future Work

• Our work is on augmenting traditional service discovery process with context awareness

• The *persistent service discovery* mechanism promises to simplify the design of clients in pervasive environments

• Future work
  – Determine network and computational impact
  – Extend existing CASD system for deployment as a bundle within OSGI runtime (to be used in AMIGO project)
  – The Context Comparator Service will use of the context ontology and the client’s query for service discovery to evaluate both the client and services contexts for aspects such as closest, fastest, cheapest etc.
Acknowledgements

- Cristian Hesselman, Henk Eertink, Sorin Iacob from Telematica Instituut
- Aart van Halteren from University of Twente
Questions?

Thanks!

Comments and suggestions welcome at

P.Pawar@utwente.nl
Andrew.Tokmakoff@telin.nl