Mediator-based Distributed Web Services Discovery and Invocation for Infrastructure-less Mobile Dynamic Systems

Abdul Haseeb\textsuperscript{1}, Mihhail Matskin\textsuperscript{1,2}, and Peep Küngas\textsuperscript{1}
\textsuperscript{1}Royal Institute of Technology (KTH), SE-16440 Kista, Sweden
\{ahaseeb, misha, peep\}@kth.se
\textsuperscript{2}Norwegian University of Science and Technology (NTNU)

Abstract

Mobile autonomous systems like robot swarms or mobile software agents operate in a dynamic environment pertaining self-organization, self-configuration and heterogeneity of computing entities. In such settings there is a need for autonomic publishing and discovery of resources and just-in-time integration for on-the-fly service consumption without any a priori knowledge of available services both within the execution environment and from the outside world. We propose a mediator-based distributed Web services discovery and invocation middleware. Moreover we present experimental results on an implemented robot swarm simulation environment. We propose a conceptual classification of computing entities on the basis of communication capabilities and conceptual overlay formation for query propagation. Our approach provides a loose coupling in terms of space and time and uses both Internet-based communication and RDF-based communication via messages mediators/post-boxes between entities when inter-communication between entities is not possible.

1. Introduction

Mobile autonomous systems like robot swarms or mobile software agents operate in a dynamic environment which pertains self-organization, self-configuration and heterogeneity of computing entities [9]. Developing an open knowledge environment for such mobile autonomous systems relies on seamless knowledge sharing, communication and collaboration with other entities both inside and outside the environment.

This puts forward requirements for autonomic and decentralized publishing and discovery of resources and just-in-time integration for on-the-fly service consumption without any a priori knowledge of services both within the execution environment and from outside world in a symmetrical way. Moreover intelligence of entities also emerges from re-using the useful functionality implemented in external world [11]. In many cases there is also an assumption of program uploads in robot swarms from external environment only [12, 13].

Interoperability issues arise as well because of heterogeneous application systems with different hardware and software execution platforms, different APIs and programming languages. Furthermore no assumption can be made about homogenous communication capabilities as well [10]. Thus we believe that there is a need to consider the integration aspect of infrastructure-less autonomous systems from a broader perspective where an interoperable and just-in-time integration solution is needed.

Platform- and language-agnostic abstraction layer as provided by Web services can provide such a mature solution for interoperability problems in such systems. Ronny et al proposed using external functionalities as Web services to serve resource-constrained devices [11]. Symmetrically reverse approaches have been proposed by Laurentiu [14] and Joshua [15].

Although Web services are generally exploited in stable client-server networks, where service registries [28] are used for discovering required services, in case of infrastructure-less dynamic environments existence of centralized control/registries can not be assumed. This in turn leads to issues regarding keeping an updated view of available services and providing reliable service discovery. Furthermore, lack of stable connectivity [31, 9] and heterogeneous communication capabilities [9] puts forward the requirement to use mediator-based communication [19].

In this paper we present a mediator-based distributed Web services discovery and management
middleware which is primarily targeted for robot swarms. Anyways, the proposed principles can be applied to any infrastructure-less dynamic environment. We propose a solution where dynamic environment and computing entities are abstracted to a P2P system and a conceptual classification of entities according to their communication capabilities is used. We assume that the physical environment has RFID tags or some other type of mediator entities, which can be used for communication when point-to-point communication between entities is not possible. Our approach provides a loose coupling in terms of space and time and overcomes the shortcomings of availability of both service requesters and service repositories at the same time [34].

The rest of the paper is organized as follows. First, in Section 2, we compare our approach to existing distributed UDDI and decentralized Web services solutions. In Section 3, we present the general communication architecture, explain entity classification and introduce basic principles of semantic Web service discovery and invocation. Section 4 explains further our approach by using an explanatory example and our realization of MTL (Message Transport Layer) [16]. In Section 5 we conclude with ongoing work and future directions.

2. Related work

One of the most compelling argument of SOA (Service-Oriented architecture) is the reuse and sharing of services between different entities. Web services (a SOA manifestation) achieve these objectives using clearly defined standards of WSDL (Web services Description Language), SOAP (Simple Object Access Protocol) and UDDI (Universal Description Discovery and Integration). Among these standards UDDI [28] provides mechanisms for service publishing and discovery using a centralized registry, a directory for publication and discovery of categorized Web services.

Traditional centralized resource management frameworks have limitations both in their fault tolerance and scalability [29]. The biggest criticism of Web services from distributed computing standpoint is its reliance on centralized service repositories [1]. UDDI is the de facto standard for Web services discovery and invocation, but tight-replication requirements among service registries, and lack of autonomous control has severely hindered the widespread usage of UDDI [26]. On the contrary P2P (Peer-to-Peer) networks [4, 5] promise high availability of published content, optimized use of network resources, content distribution cost and scalability using decentralized architectures.

Towards synergizing P2P networks and Web services, Boualem et al [21] proposed a declarative dynamic composition and execution framework for Web services in P2P networks. But their work did not consider the Web service publishing and discovery in P2P networks, which is the primary aspect to overcome the scalability issues.

Sujata et al. [25] proposed DUDE (Distributed UDDI Deployment Engine) to address the scalability issues with UDDI. DUDE proposed the leveraging of structured DHT (Distributed hash table) [4], a P2P system that forms a structured overlay, allowing more efficient routing than the underlying network, as a rendezvous mechanism between different registries. In their approach service description message dispersion to several distributed UDDI registries promote scalability and replication. But such an approach can not cope with dynamism, mobility and scenarios where inter-communication between entities is not possible in a point-to-point fashion. Similarly Zhenqi et al [22] proposed a Web services architecture based on a P2P network (built on top of JXTA). They proposed a classification of peers on the basis of computation power and memory into Service-peers and Super-peers, with Super-peers responsible for publishing of Web services, query routing and formation of peer-groups for Service-peers. Such an architecture assumes homogenous communication capabilities of peers, which can not be considered in heterogeneous environments as robot swarms. Moreover we do not consider a full coverage, of an entity, of rest of the network and mobility of entities can restrict the communication coverage of an entity.

Towards establishing a federation of UDDI registries into different business domains or communities, Emil et al [24] proposed an approach for abstract meta-data based classification of community registries with one root/centralized registry for communities. In a similar approach Mike et al. [30] proposed a federated architecture for P2P Web services, in which a federation for UDDI-enabled peer registries is employed in a decentralized fashion. Publishing of Web services was done on centralized UDDI peer registries and then peers join service syndication. Such federated approaches suffer from single point of failure.

Zongxia et al. [27] proposed the notion of an active and distributed service registry (ad-UDDI), an active monitoring mechanism enabled UDDI to maintain periodic service information. They considered a layered approach for distributed UDDI registry into a management root layer, a business layer comprising of domain specific ad-UDDI registries and a service layer.
Towards infrastructure-less environments, Zakaria et al. [31] proposed mechanism for dynamic management of distributed UDDI in the absence of wired communication infrastructure. They introduced the notion of Messenger (a mobile user with a user-agent) which, upon reaching the vicinity of some UDDI, updates the Web service descriptions it has cached already for user needs. Exchange of information depends upon ad-hoc nature of network and occurs when a user enters a new cell i.e. a network coverage area. Proposed environment of Zakaria et al. [31] suits best the requirements of poorly reliable and unpredictable coverage feature of communication infrastructure, but their approach did not consider the use of mediator entities [19] and moreover message routing aspects were not taken into consideration.

3. Mediator-based distributed semantic Web services middleware

Web services abstraction layer allows applications and entities to communicate in a programming language- and platform-agnostic manner and has developed into a mature solution for providing interoperability between heterogeneous computing entities. Peer services [20] and Web services are the most referred technologies for decentralized applications, which apparently address different problem domains yet have a significant conceptual overlap.

Web services provide a just-in-time integration solution for on-the-fly service consumption without any a priori knowledge of the service and procedures a consumer needs to reuse. But the biggest criticism of Web services from distributed computing standpoint is its reliance on centralized service repositories [1]. Peer services [20] on the other hand negate the notion of centralized repositories in SOA paradigm, though also leverage SOA but synergize producer and consumer as peers and solves the centralized nature of Web services by using cooperation of peers [2]. Web services have clearly defined standards in form of WSDL and SOAP. While JXTA considers a more abstract approach yet depending upon exchanging XML based messages.

3.1. Basic Philosophy

Our proposed communication middleware [16] considers the basic philosophy of JXTA (project juxtaPose) [2], a P2P stack and XML based set of peer platform protocols and fundamental services for discovery, presence management and communication between peers. XML-based protocols allow any device connected to network to exchange messages and collaborate independently of underlying network topology thus making XML based JXTA transport- and language-agnostic. Apart from XML based communication the introduction of unidirectional asynchronous communication channels (Pipes) is one of the basic technology shift in JXTA. Asynchronous and nondeterministic nature of network contributes significantly towards high availability of published content, services, optimized use of network resources, content distribution cost and scalability.

Our basic philosophy of communication middleware takes the communication principles from JXTA and uses the defined standards of WSDL, RDF and SOAP to achieve interoperability. We propose an abstract approach of classification of computing entities on the basis of their communication capabilities. In principle there are different types of protocols depending upon the communication capabilities of computing entities.

As compared to JXTA our middleware not only relies on communication capabilities on entities, but also due to dynamic nature (caused by mobility of entities) it supports dynamic joining of entities in different groups during system execution.

3.2. Conceptual classification of Entities

Rendezvous computing entities (RCE) are the computing entities with wireless connectivity or entities which can communicate outside a domain or with external world such as Internet. Rendezvous entities provide efficient dissemination of requests from one domain/group to the another. Thus they can be considered as routing dissemination of requests from one domain/group to the another. Thus they can be considered as routing peers [4, 5, 6]. In a way, rendezvous entities create a virtual overlay network with other rendezvous entities and serve as distributed points of centralization of network, serving entity-groups [22] at a lower level.

Edge computing entities (ECE) are considered those entities that don’t have a capability to communicate in a point-to-point fashion with other computing entities and require some sort of information mediator/relay to communicate their messages. RFID tags could be one example of message relays, that could serve ECE (if ECE are only equipped with RFID read-write capability and no other point-to-point communication capability with other entities).

Another example is distributed knowledge bases, each knowledge base can be abstracted as a message relay for those software agents that can only communicate with in domain of that knowledge base, while some software agents can have an additional...
capability to serve as a gateway and communicate to other distributed knowledge bases by communicating with respective gate-way agent.

RFID tags are seen as a special example of MR, which are being employed in numerous robotic environments. In general, RFID tags have limited memory, thus storing an XML based message is not a viable solution. For such memory-constrained MR, we consider the communication via extended RDF block rows data model [16]. Apart from typical (Subject, Predicate and Object) we consider some additional fields such as time of data insertion/update, source of data, context field etc. (see Section 5).

3.3. Communication middleware architecture

Modular architecture of our communication middleware (CMA) is depicted in Figure 2. Detailed elaboration can be found in [16]. Message Transport Layer (MTL) provides an implementation of asynchronous communication channels identified by endpoints, similarly to normal Web services invocation. An entity can logically bind itself to any MTL of destination entity, which can bind itself further to subsequent destination MTL(s), thus creating a chained communication overlay network. Local Service Registry (LSR) serves as a local cache for Web service discovery. For LSR we have implemented a light-weight UDDI [16]. An incoming service request is first searched from LSR before being cached to Query Response Cache (QRC). QRC caches all incoming Web service requests and propagates back response messages. Entity Discovery Registry (EDR) serves as a record for discovered entities. EDR is used to create a semantic topology based on set of services an entity advertises and provides semantic query propagation (see Section 3.5). Ontology repository stores entities’ ontologies in form of RDF triples which are used for mapping Web service discovery queries to published Web services. Local Robot Knowledge Base (LRKB) serves as a knowledge base of factual information, which an entity might gather during its execution cycle. LRKB and the associated defeasible reasoner are specified in [9, 18]. Interested readers can read detailed discussed about the architecture, different modules and our defeasible reasoner implementation in [16, 17, 18].

3.4. Entity and Web service registration

Upon a system bootstrap and encountering a MR, entities register themselves by expressing their presence information using MR’s data model (in case of RFID using an extended RDF data model). When registering at a MR, entities also receive a list of other registered entities from the respective MR. Registration also includes advertising Web services descriptions and a set of entity’s expertise. For instance expertise can be Weather Information Services, Cleaning Services etc. depending upon the kind of entity and services an entity might provide.

Upon discovery of some rendezvous entity by another rendezvous entity, each rendezvous entity initiates the Rendezvous Peer View (RPV) protocol to establish a RPV, which is like creating a finger table pointers [4] for inter entity-group discovery and communication. RPV protocol involves periodic sending of a random list of RPV to one of the discovered rendezvous entities via the Internet. In a similar fashion each rendezvous entity sends a periodic RPV and purges non-responsive rendezvous entities from its RPV.

Effectively each RCE maintains an entity-group around a MR and serves for dissemination of messages from one entity-group to another using the Internet communication by using RPV. MRs at the same time are responsible for dissemination of messages inside a particular entity-group. The registration process is illustrated in Figure 1.

3.5. Entity selection via semantic topology

In such an abstraction to P2P architecture, each entity operates with identical functionality. Such an abstraction features redundancy, dynamic selection of entities, and fault-tolerance. Despite these benefits, the inherent difficulties related to P2P technology, which
need to be solved, are propagation of query to an appropriate entity and efficient routing of messages.

Several researchers [4, 5, 6] have tried to solve the efficient dissemination of messages from one entity to another while Berners-Lee et al. [7] advocated on the enrichment of information with well-defined meanings to enable computing entities to collaborate by expressing the knowledge in well-defined formal way. Notion of RCE, RPV and dissemination of messages from one entity-group to another provides quick dissemination of messages across entity-groups, while MR performs dissemination of messages within an entity-group.

To incorporate semantics and to perform semantic-based query propagation to appropriate entities, we use a shared ontology of computing environment and entity expertise in the network. For enabling this we have created a semantic topology on the basis of semantic similarity of expertise of entities and subject of query. Subject is an abstraction of query expressed in terms of shared ontology of environment. Each entity along with service description, also specifies its set of expertise. RCE maintains the expertise of the entity-group (referred to as expertise tuple) which is built-up by the registered entities’ expertise set at a particular MR.

We create virtual semantic topology by each entity recommending a requesting entity to another entity whose expertise is semantically similar of subject of query. Based on recommendation, requesting entity can re-query after discovering the availability of semantically similar expertise in the network. Semantic topology is created by each entity caching the services of entities whose expertise are semantically similar to its own expertise.

In our current implementation, the similarity function for mapping the subject of query to expertise of an entity relies on semantic distance of subject of query to the expertise an entity. In particular our semantic topology is created by using parent-child relationship. We are planning to incorporate a more general similarity function [8] for mapping subject of query to expertise description.

Shared ontology of environment/domain is expressed as RDF triples in Ontology Rep for describing the functional expertise of entities. As elaborated earlier, entities advertise their Web services as advertisements, when coupled with their set of expertise, it provides a decentralized solution to advertise Web services. Ontology Rep is transformed to a set of facts and rules in LRKB and defeasible reasoner is applied. Currently we have a light-weight implementation of defeasible reasoner [18] which we are using for formal concept/ontology learning. We express similarity function as a set of rules in defeasible logic as well. The purpose of developing a rich set of rules for communication, concept learning and associated semantic similarity is to provide a rule-based expert system, using which both communication and reasoning capabilities of entities can be made intelligent [18].

### 3.6. Distributed Web service discovery

Decentralized design of Web services publishing and discovery is more scalable and reduces the overhead of centralized updates. Moreover, it provides higher degree of fault-tolerance as compared to existing centralized architecture of UDDI [29].

As shown in Figure 1, ECE and RCE publish their Web service descriptions and sets of expertise via MR. To implement this step in our middleware first a WSDL document is encoded to suitable data-model for MR. In addition to MR encoding, our current implementation also provides an encoding/decoding of XML/SOAP to extended RDF data model to be stored on RFID tags. Instead of using a hash function to store service descriptions on ECE or RCE we consider the following approach.

1. RCE maintains all Web service descriptions and expertise sets for all the entities associated with a particular MR (expertise tuple). Service descriptions of an entity-group are not advertised by RCE across other entity groups, instead RCE uses RPV protocol to share the expertise tuple of its respective entity-group with another RCE.

2. ECE maintains only the Web service descriptions from those entities whose expertise are at a semantic distance of at most 1 from its own expertise (Degree of separation of two concepts is 0 if both are identical concepts. If there exists a parent-child relationship between two concepts then degree of separation is 1 and so on).
Any request for a Web service discovery from ECE is registered at respective MR, which can be read by any other entity. When an entity receives a query, it first checks its LSC. If it can find a mapping of query’s subject with expertise of some entity from its LSC, then it will provide a recommendation as per its LSC about the appropriate target entity, i.e. entity will advertise about the entity which has an expertise which is semantically similar to query’s subject (see Figure 2). The requesting entity can take following action:

1. Entity can re-query after discovering of availability of a semantically similar expertise in the network (if an exact match is not available in network)
2. Entity can trigger discovery protocol to query the service descriptions of the entity which has been recommended.

Similar principle is followed by RCE. They can forward the query to another RCE upon matching the subject of query with expertise tuples of target RCE.

4. ROBOSWARM Case Study

We apply the concepts and principles described in Section 2 and Section 3 in the ROBOSWARM project [9]. The motivation of ROBOSWARM project is to develop an open knowledge environment for self-configurable, low-cost and robust robot swarms operating in dynamic environments. The swarms of robot are attributed by heterogeneous communication capabilities - i.e. few robots may be equipped with a wireless capability, while the other robots, in the absence of wireless capability, have to use RFID tags for reading/writing data.

In this project we intend to create and maintain a distributed data environment, based on intelligent inter-robot communication, and a global robot database solution for knowledge reuse. Thus communication interoperability of robot swarms both with environment entities and outside world is needed. We use Web services for exposing robotic functionality and abstracting each robot to a Web service interface. A Web service based robot control platform using RFID tags as information mediators has been proposed previously [19] by Bon Keun et al.

Apart from LRKB of each robot, we also consider a Global Robot Knowledge Base (GRKB) which refers to collective swarm knowledge [17]. GRKB runs on a server, equipped with wireless connectivity, which can perform higher level reasoning and can provide resource-intensive services to swarm entities. Also it can serve as a gateway entity to swarm while communicating with outside world. In principle any swarm entity with wireless connectivity can serve as swarm gateway.

Using the entity classification as proposed earlier, we classify robots on the basis of their communication capabilities. Apart from robots, there are a number of RFID tags in the operational environment which serve as information mediators. We classify the swarm entities as follows:

- **Rendezvous robots (RCE):** The robots with wireless connectivity,
- **Edge robots (ECE):** robots that do not have a wireless connection and require some relay entity to relay their service request(s) or response(s) or user-level messages,
- **Message Relays (MR):** Effectively RFID tags.

We use extended RDF data model encoding for RFID tags.

![Figure 3: Rendezvous robots overlay formation & communication b/w Edge Robots & RFID](image-url)
Operational scenario of ROBOSWARM with entity classification is shown in Figure 3. Control messages from UI can be disseminated into Swarm either by using wireless interface provided by rendezvous robots or by writing the RDF encoded message to RFID tags. Similarly GRKB can interface to either RFID tags or to rendezvous robots. Rendezvous robots create an overlay network and run RPV protocol to maintain a RPV and robot-group expertise tuples. Lower level communication between edge robots and RFID tags comprises of Web service advertisements, expertise advertisement, request and response message and miscellaneous control messages. This lower level communication uses extended RDF data model [16].

5. Experiments and Conclusions

For experimentation of our proposed middleware we use Player/Stage Simulator [32]. Player/Stage Simulator provides a client control program that can talk to player interface of robot actuators and sensors over TCP Socket. Our current implementation of middleware (in C++) is small and efficient (15KB memory footprint).

A brief snapshot is shown in Figure 4. For the sake of simplicity of presentation, we only show two mobile robots with Wi-Fi communication capability (i.e. Rendezvous robots, represented by Blue robots, with the enclosing circle representing their Wi-Fi range). Red robots represent edge mobile robots which are only capable of communicating via RFID (message relays, which are expressed by brown entities in simulation). The communication infrastructure modeled for experiments is attributed by instability and partial communication coverage due to robotic mobility. Thus robots can get out of coverage area of Wi-Fi or read/write range of static RFID in environment.

Web service descriptions are expressed as RDF. The reason for selection of RDF is to leverage the simplicity and semantic richness of RDF. RDF based Web service description has already been used by Brahmanadna et al. [23]. In our current implementation MTL provides a set of API for Web service registration which are invoked by a robot client. Robot client automatically transforms the Robotic proxies (e.g. PlayerCc::Position2dProxy, PlayerCc::WiFiProxy) expressed by Player drivers into RDF based Web service description as well to achieve the goal of exposing a robot and its all interfaces as a Web service.

RDF encoded knowledge base at RFID (see Figure 4) expresses the presence and registration of an edge robot with ID 7010. Service registration by robot is func_1. The context field of RDF sextet refers to interpretation of RDF sextet for instance: reg (register), pub (publish), query (query a Web service), invoke (invoke an interface) etc.

In our implementation of middleware the discovery of a Web service is done on the basis of keywords and expertise of an entity is used for efficient entity selection for query propagation. But there is a need for formal specification of semantics for service description in RDF as well. In our ongoing work we are using the approach of [33] to model the real-world concepts related to a Web service and map the ontological concepts with WSDL. The semantic model and associated mapping will then be stored in LSR as RDF. Thus a entity communicating with another entity, or a request for a Web service can be inspected against

Figure 4: Simulation Snapshot & extended RDF encoded knowledge with 2 rendezvous robots, 2 edge robots and 2 relays
the ontological concepts. In a broader perspective, we intend to create a rule-based service discovery in which rules can provide the foundation for semantic reasoning and formal concept learning [18] to overcome the limitations of using a pre-defined ontology of domain.

We are also extending similarity function [8] for query’s subject mapping to expertise description. Also we are experimenting with an algorithm for adaptive degree of separation based on network topology to cache services for the creation of a flexible semantic topology.

Acknowledgements
This work is partially supported by a European 6th Framework project - ROBOSWARM.

6. References