

Geminga: Service Discovery for Mobile Robotics

Philipp A. Baer, Thomas Weise, and Kurt Geihs

Distributed Systems Group
Kassel University, Germany

baer,weise,geihs@vs.uni-kassel.de

Abstract

Communication infrastructures of highly distributed software architectures are often complex to configure and costly to manage. Dynamics in the environment leading to a need for self-adaptation will further complicate maintenance. For this purpose we introduce Geminga, a service discovery approach for application in ad hoc networks and self-configuration of communication infrastructures. Tailored to unreliability in communication, it bridges the gap between traditional application domains of distributed architectures and groups of mobile robots. The applicability of Geminga is shown by adapting it to a software framework for autonomous soccer robots.

Preview

This document is a preview version and not necessarily identical with the original.

<http://www.it-weise.de/>

1 Introduction

Communication infrastructures of highly modular distributed software architectures are often complex to configure and costly to manage. This is especially true for architectures where the configuration is dynamic and frequently changing. Autonomous mobile robots acting in dynamic environments are one example for such a scenario which gained more and more importance in recent years. Spica, introduced in [1, 2], is a sophisticated model-driven development framework for mobile robotics which promotes a modular, service-oriented, and platform independent soft-

ware development approach for communication infrastructures.

Typical tasks of autonomous mobile robots involve negotiation of tactics, assignment of roles, and exchange of environmental information. Each of these tasks is represented as a service in a Spica-based robotic software architecture. As services represent self-contained software structures, they are free to request further information from other sources and can even provide new information source themselves. Service-orientation not only leads to increased modularity but also addresses heterogeneity issues. Different types of robots will provide different services and will thus be equipped with different sensors and actuators. A rescue scenario, for example, might involve airborne robots as well as ground machinery. For robotic soccer, a quite homogeneous but specialised hardware platform will be used. The structure and approaches followed for cooperation and collaborations in these scenarios differ fundamentally.

In this paper we introduce Geminga, a multi-purpose service discovery and matching approach for mobile robotics that complements communication infrastructures generated with Spica. Each participating system runs one instance of Geminga, allowing service providers to announce the availability and the interfaces of their services. In turn, local software modules that request certain service functionality are informed about its availability, covering all services available in the network. Geminga does not establish communication channels by itself, but triggers their creation between service consumers and providers in the service.

1.1 Service Discovery in Mobile Robotics

Geminga is designed for mobile robots which are connected via wireless networks. Such media are known to be vulnerable to failures whereby mobile robots may get unavailable or leave a group unattended at any time even during controlled operation. This may also be caused by physical effects or system failures. Hence, measures have to be taken to make dynamic negotiation processes robust.

Adaptation capabilities have to deliver reliable results, even when used in such unreliable environments. Simplicity in design and protocol interaction is known to contribute to the robustness of a communication approach. Below, we propose a list of some desired functionality for Geminga.

- **Distributed Operation.** No central knowledge repository or discovery service can be employed because otherwise, discovery is vulnerable to network unavailability. The approach should furthermore consume low bandwidth and therefore involve only sparse protocol interactions. It must be robust against network failures.
- **Passive Operation.** Service consumers should be able

to listen to the communication in the network in a minimally invasive manner without influencing service providers in any way.

- **Scalability.** The system should scale up to about 30 participants. This number is derived from the average size of a group of mobile robots including control and monitoring instances. In our case about ten robots and an arbitrary number of additional clients are involved.
- **Self-Management.** The proposed architecture should be self-configuring or require only minimal configuration effort.
- **Semantic Matching.** The approach should be kept generic and allow matching of semantic service types.

This paper is organised as follows. In Section 2, we discuss related work in the domain of service discovery. Section 3 then introduces Geminga, our approach for service discovery in mobile robotics. A brief discussion of results obtained during an experimental evaluation is shown in Section 4. We conclude the paper in Section 5 and point to future work.

2 Related Work

Service discovery in computer networks is a broad domain. There are many different approaches which target discovery, integration, and configuration of services or hosts. Robust service discovery schemes for unreliable or ad hoc networks are, however, still subject to active research.

In *Service-oriented Architectures* (SOA), service discovery plays an important role for automated service composition [3, 4, 5, 6, 7]. The *Universal Description, Discovery and Integration* (UDDI) [8] resembles a standardised directory service which realises one possible service discovery approach for SOA architectures. Being a centralised approach, it is not considered any further in this context.

Zeroconf [9] comprises a set of technologies facilitating automatic network configuration and discovery of services and devices. Well-known implementations are *Bonjour* by Apple and the open source project *Avahi*. A Zeroconf architecture builds on an extension of the *Domain Name System* (DNS) by embedding service description records (DNS-SD) [10]. Multicast-enabled DNS (mDNS) [11] simplifies DNS queries in local area networks by allowing multicast queries.

The UPnP working group specifies an automatic network configuration approach very similar to Zeroconf. It builds on the *Simple Service Discovery Protocol* (SSDP) [12] which itself relies on HTTP.

Zeroconf lacks the flexibility required for dynamic robotic systems whereas UPnP depends on HTTP and TCP, violating the previously mentioned bandwidth constraints.

The *Service Location Protocol* (SLP) [13] provides service discovery for local area networks. Devices joining the network use multicast for initial service discovery. Afterwards, the transmissions still are mostly message-oriented. Actors in an SLP-enabled infrastructure are *User Agents* (searching for services), *Service Agents* (providing services), and, optionally, *Directory Agents* (offering caching functionality). SLP is designed for reliable communication and thus not feasible for dynamic ad hoc systems.

In [14], Lenders et al. present a service discovery approach modelled after electrostatic fields. It is lean, robust, and completely decentralised. It resembles a form of publish/subscribe system where processes can subscribe to messages containing information on specific subjects. The system explicitly targets mobile ad hoc networks without routing capabilities and therefore introduce own routing functionality.

Geminga follows a similar approach but relies on the routing capabilities provided by the underlying network. It can thus do with fewer messages and less protocol interactions.

Elvin [15] is a distributed event routing and delivery service with publish/subscribe semantics, featuring a sophisticated content-based subscription system. Subscriptions are formulated using an Elvin-specific requisition specification language and content is specified as key-value pairs. Elvin installations comprise a central router component, several of which may be federated. Being a sub-concept of content-based subscription, the type-based subscription approach of Geminga is more suitable for the application domain we are dealing with.

3 Geminga Distributed Service Discovery

The pulsar *Geminga* in the constellation Gemini, closer to Earth than any other known pulsar, is the inspiration for the name of our service discovery approach. The *Geminga Service Discovery* is developed in conjunction with the Spica [1, 2] development framework for autonomous robot software. More specifically, Spica supports developers in creating the communication infrastructure required for a distributed, modular software architecture. To allow such an infrastructure to automatically configure and adapt itself to changes in the network configuration, a robust service discovery is required. In the following, we will refer to instances of Geminga as *participants* and to the modules of the robot software interacting with them as *service peers*, since they act as service provider, service consumer, or both.

Addressing the requirements worked out in Section 1 and taking into account the related work analysis, we use

a beacon-based service discovery approach similar to [14]. Each participant announces its published and subscribed services on a regular basis, providing the other participants with a snapshot of its local state. This announcement flooding introduces an upper boundary for scalability which, however, is well above the specifications stated in the requirements.

Each participant maintains a local data structure for storing received beacons. Outdated entries are automatically purged after a given amount of time automatically. Service matching is based on the locally stored announcements and will be discussed in Section 3.6 in detail.

Creation or removal of communication channels between the service peers is triggered by Geringa at the respective peers. Such channels may operate in any of the three modes shown below.

- **One-to-one (oto).** One-to-one channels are configured and established only once between two or more participants. No information exchange is needed.
- **One-to-many/stream (otm/stream).** Channels configured for otm/stream mode are set up and established passively. They resemble sources for streaming data with more than one recipient which just have to listen at specified addresses. Information exchange is required from providers to consumers only.
- **One-to-many/pubsub (otm/pubsub).** Channels configured for otm/pubsub mode are established as soon as at least one subscriber is present. They are shut down after the last consumer has withdrawn its subscription. Bidirectional information exchange is required between the service peers.

3.1 Geringa SD Layout Decisions

Geringa SD is a core component of the Spica software development framework, but is designed in a very general manner which allows for adoption in arbitrary platforms. Service types are identified using generic *identity* tokens which can either be 32bit unsigned numbers, text strings, or semantic identifiers consisting of a *concept* and a *representation* [16]. Using a hash algorithm, numerical identifiers can be derived from their semantic counterparts.

The service peers on each host access their local Geringa instance as clients in order to publish or request services. This interaction relies on the *Geringa Client Negotiation Protocol* (GCNP) as defined in Section 3.2.

Announcement messages are distributed among all Geringa instances, propagating the locally available services as well as service subscriptions. The discovery process as such does not involve any network communication, since it is based on the aggregated information from other participants and the information received from local service peers

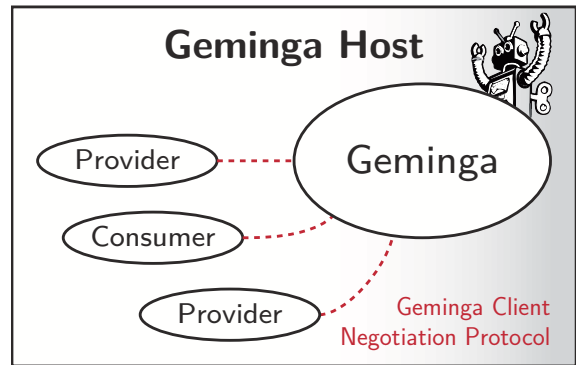


Figure 1: Geringa with service peers

only. The approach thus becomes insensitive to networking issues. Figure 1 illustrates the layout of a Geringa installation on a single host with three service peers – two providers and one consumer – connected via GCNP.

3.2 Geringa Client Negotiation Protocol

GCNP is a simple, text-based protocol used for information exchange between service peers and the local Geringa instance. For this (strictly local) communication, TCP/IP connections are used.

Geringa informs clients of the availability of a service as well as subscription events from consumers via GCNP. It does not create any connection, though. Instead, communication channels between service peers have to be established by the peers themselves. In order to ease the integration with existing software packages, interface stubs implementing the GCNP are provided for several programming languages. Dynamic channel creation and removal as well as management tasks are all handled by these interface stubs in a transparent manner.

3.2.1 Protocol Operation

The structure of all statements exchanged between Geringa and its service peers obeys the simple grammar below:

```
<id> <command> <subcommand> <parameters>
```

Each statement is identified by a 32bit unsigned integer *<id>*. If it equals 0, the statement is either directed at Geringa (*system command*) or received from it (*system response*). All the remaining values are used to establish a relationship between statements, i.e. between a command and a response or event. Parameters are collections of key-value pairs delimited by spaces. The value part may be omitted. Each message is terminated by a CR+LF sequence.

GCNP currently supports four types of statement, namely *messages*, *commands*, *errors*, and *events*. We will

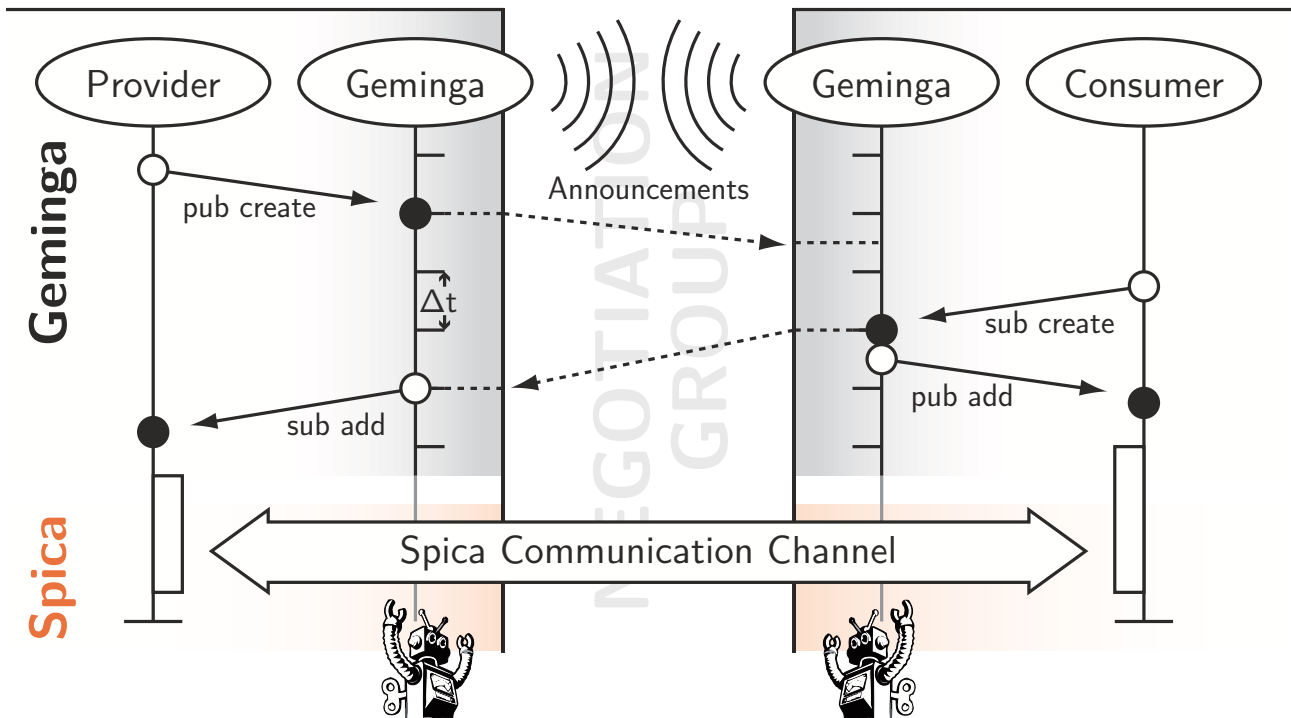


Figure 2: GCNP interaction and establishment of a Spica channel

shortly outline the characteristics of commands and give an example for the protocol interactions of GCNP in Figure 2.

3.2.2 Messages

Messages are generated by Geminga to return either information concerning the Geminga instance or as response to a previously issued system command. Here, `<command>` specifies the message type whereas `<subcommand>` may be used for specialisation. Parameters are completely optional.

3.2.3 Requests

Requests are issued by service peers to trigger execution of operations at a Geminga instance. No immediate responses are generated (except in case of an error). Instead, responses are delivered asynchronously. The commands for service publication and subscription are shown below, covering announcement, modification, and removal of services.

The `pub` command allows service providers to configure service offers. Subcommands are used to trigger creation (`create`), modification (`modify`), or removal (`remove`) of an offer. The service itself is characterised by a set of parameters which includes the channel mode and the service identity, for example.

The `sub` command allows service consumers to configure a service subscription. The subcommands and parame-

ters are identical to their counterparts in the `pub` command.

3.3 Events

In response to publication commands, subscription events inform the provider and consumer of changes in their requests. Events build the foundation for adaptive behaviour and thus resemble one of the most important features of Geminga. The structure of an event is similar to that of the corresponding request command. The subcommands, however, may differ.

A service subscription will generate a publication event (`pub`) with the respective subcommand once a service appears (`add`), disappears (`del`), or changes its configuration (`modify`). The parameters represent the configuration of the service.

A service publication will lead to a subscription event `sub` with the respective subcommand. The subcommands and parameters have the same semantics as for publication.

3.4 Errors

In case of an error while executing a command, error messages are sent back synchronously. The command element of the original message is replaced by a triple-digit error code, the subcommand and parameters are replaced by a descriptive string.

3.5 Announcement

Geminga SD follows the simple yet powerful approach of periodical service announcement. Waiver of elaborate protocol interaction schemes that handle network failures is compensated in our approach by providing redundancy in data transmission. The repeated transmission implicitly provides this redundancy. Publications and subscriptions are transmitted in the same announcement message, effectively reducing the number of required messages thanks to piggybacking.

3.5.1 Geminga Identifier Generation

Before introducing the announcement scheme, we will outline the generation of unique identifiers for Geminga instances. These identifiers are needed because every host – and hence, each instance of Geminga – potentially has several valid network addresses which thus disqualify as unique identifiers.

First, a random number is generated and broadcast to the other Geminga instances. If another system already owns this number, the process is repeated until a unique identifier was found or an iteration limit has been exceeded. In the latter case, Geminga shuts down.

If an identifier clash is detected during runtime, all involved participants have to drop their current identifier and generate new ones using the procedure outlined above. This will force all Geminga instances to temporarily cancel all service publications and subscriptions which are then renewed after a new identifier was found.

3.5.2 Service Announcement

By default, services are announced every two seconds. It is possible to automatically adapt the interval to the number of participating Geminga instances or the relative message loss. The latter one is accomplished by monitoring the incoming traffic and interpreting its distribution.

Figure 3 outlines the format of a Geminga announcement message. The message header block contains management information required for processing. It will not be further discussed here. The address block contains a list of all local interface addresses. They are collected during start-up and required for establishing communication channels. For each available interface, a unique announcement message is generated, containing only addresses and services that are valid for this specific interface. For loopback devices, no announcements are generated.

The publish and subscribe block lists the local service peers (providers and consumers) along with their publications and subscriptions. Each peer is listed with its name and a 32bit unsigned integer number which acts as a unique identifier within the Geminga instance.

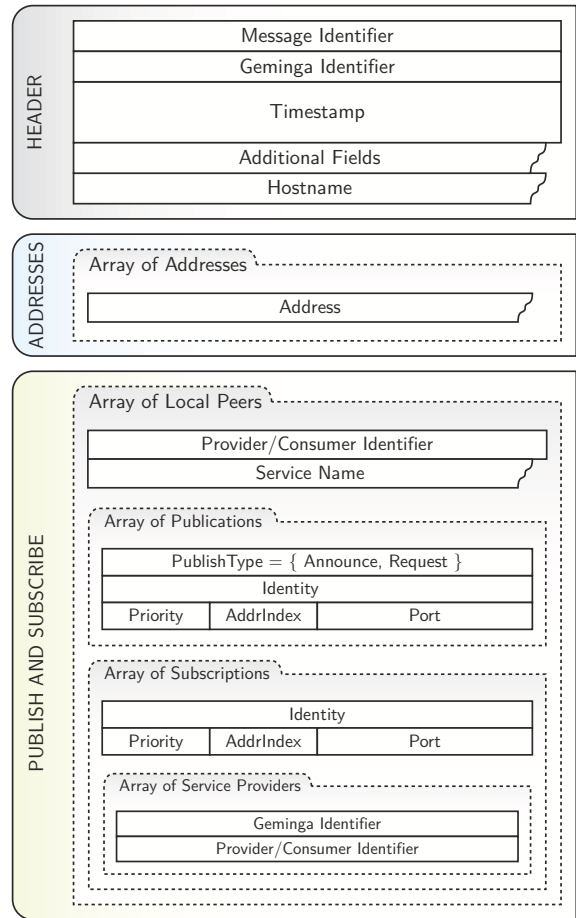


Figure 3: Geminga announcement message

A peer description contains two lists. The first one describes service publications along with the channel mode, the service identity, a priority, and the address index/port tuple. The priority field is required in the matching process for arbitration. The second list describes service subscriptions along with the consumer identity, a priority and address data. A subscription further contains a list of service provider identifiers (Geminga and peer identifiers). This enables service consumers to specify their favoured providers to subscribe to.

This message structure, along with a space efficient encoding scheme, allows for reasonably sized announcement messages even if dozens of services are involved. Specified in SpicaML/MDL, the Spica modelling language for message formats [2], the message structure is generated automatically for all required platforms. It is further possible to authenticate, encrypt, and compress the message.

3.6 Matching

Geminga instances maintain a data structure for the most recent announcements received from the other systems. Based on these announcements, the matching process can be executed locally. Each system therefore listens to the announcements exchanged in the negotiation group and carries out extensive matching operations on the local host without disturbing other participants. The stored announcements are indexed in several ways, each representing a search criterion required for the matching process. The most important indices are outlined below.

The indices on service publications or subscriptions consist of five fields which uniquely identify a service description: the service peer and Geminga identifiers, the service identity, the channel mode, and the service address. Addresses are compared based on scope equivalence.

If a service consumer requests certain functionality, the collected announcements (including the locally available services) are searched. Depending on the number of search results, an arbitration step is carried out, based on the priority specification.

Once a match was found, a GCNP event is triggered for the respective local peer. In case of modifications of already processed service requests, a modification event is generated respectively.

For each service, Geminga maintains a timestamp of the last update and an index thereon. Service request and offer entries not updated for a given amount of time are removed from the data structure. Hence, inaccessible services or outdated subscriptions are invalidated automatically. Peers are sent a GCNP event, informing them of the removal of the service or subscription.

Since service providers and Geminga instances are not guaranteed to always sign off cleanly but rather may quit without notification, the automated invalidation is a very important feature.

4 Results

The software architecture of the Carpe Noctem soccer robots of Kassel University¹ is based on a Spica-generated communication infrastructure. Using Geminga, we were able to create a highly modular, distributed, and self-configuring software architecture. Provided services are discovered and integrated automatically with no additional configuration expenses.

The Geminga approach has been evaluated experimentally during the RoboCup German Open championships 2008 in Hannover, Germany. Geminga was responsible for negotiation and adaptation of communication links whereas

the Spica-generated communication infrastructure handled the message distribution.

Each of the six Carpe Noctem robots typically executes six independent modules. These are connected one another and to four other modules located in monitoring systems outside the playing field. Within a robot, nine services are provided. Each service generates messages at 33Hz, resulting in about 297 messages per second. In addition, each robot subscribes to seven external services which typically send messages with a frequency between 1Hz and 10Hz. About 50 messages are received per second, including messages from the five other robots. Finally, each robot provides about six further services to external subscribers. Most of them exhibit a message rate of rarely more than one message per second. One service distributes the robot's internal state with a message rate fixed to exactly 10Hz. This results in about 20 messages per second for out-bound traffic. The robots successfully competed with the other teams and finished 4th in the tournament.

These tests have shown that the Geminga approach was able to automatically establish communication links and manage the communication infrastructure in a resource efficient and reliable fashion. Even during periods of network instability the communication infrastructure configuration persisted.

5 Conclusions and Future Work

Geminga is a simple yet powerful service discovery scheme based on a beacon-oriented service announcement approach. It explicitly dispenses with elaborate negotiation and notification protocols in order to be less sensitive to networking issues. This renders the Geminga service discovery especially useful for application in unreliable communication networks.

Each host communicating with other Geminga-enabled hosts executes a single Geminga instance. Implementing a client-server architecture, Geminga allows local service providers and consumers to connect and use its services through the Geminga Client Negotiation Protocol (GCNP). This includes the discovery, matching, and management of services.

In the near future, the semantic matching capabilities of Geminga will be developed further. Therefore, support of semantic identifiers will be extended to integrate ontology lookups including reasoning. Research will be further conducted on arbitration techniques for selecting and merging multiple data sources.

References

- [1] Philipp Andreas Baer, Roland Reichle, Michael Zapf, Thomas Weise, and Kurt Geihs from the

¹<http://carpenoctem.das-lab.net/>

- University of Kassel, FB-16, Distributed Systems Group, Wilhelmshöher Allee 73, 34121 Kassel, Germany. A Generative Approach to the Development of Autonomous Robot Software. In *Proceedings of 4th IEEE Workshop on Engineering of Autonomic and Autonomous Systems (EASE 2007)*. IEEE, March 26-29, 2007, Tucson, AZ U.S.A. ISBN: 0-7695-2809-0. doi:10.1109/EASE.2007.2. Online available at <http://www.it-weise.de/documents/files/BRZWG2007AR.pdf> [accessed 2008-10-20].
- [2] Pedro Lima, editor. *Robotic Soccer*. I-Tech Education and Publishing, Vienna, Austria, December 2007. <http://s.i-techonline.com/Book/Robotic-Soccer/ISBN978-3-902613-21-9.html> [accessed 2008-10-20].
- [3] Thomas Weise, Steffen Bleul, Diana Comes, and Kurt Geihs from the Distributed Systems Group, FB 16 – Elektrotechnik und Informatik, University of Kassel, Wilhelmshöher Allee 73, 34121 Kassel, Germany. Different Approaches to Semantic Web Service Composition. In Abdelhamid Mellouk, Jun Bi, Guadalupe Ortiz, Dickson K. W. Chiu, and Manuela Popescu, editors, *Proceedings of The Third International Conference on Internet and Web Applications and Services, ICIW 2008*, pages 90–96, June 8–13, 2008, Athens, Greece. IEEE Computer Society Press, Los Alamitos, CA, USA. ISBN: 978-0-76953-163-2. Library of Congress Control Number: 2008922600. Product Number: E3163. BMS Part Number: CFP0816C-CDR. Online available at <http://www.it-weise.de/documents/files/WBCG2008ICIW.pdf> [accessed 2008-10-20].
- [4] Thomas Weise, Steffen Bleul, Marc Kirchhoff, and Kurt Geihs from the Distributed Systems Group, FB 16 – Elektrotechnik und Informatik, University of Kassel, Wilhelmshöher Allee 73, 34121 Kassel, Germany. Semantic Web Service Composition for Service-Oriented Architectures. In *Proceedings of IEEE Joint Conference (CEC/EEE 2008) on E-Commerce Technology (Tenth CEC'07) and Enterprise Computing, E-Commerce and E-Services (Fifth EEE'08)*, pages 355–358, 2008. doi:10.1109/CEC/EEE/2008.69. In proceedings [17]. Online available at <http://www.it-weise.de/documents/files/WBKG2008SWSCFSA.pdf> [accessed 2008-10-20].
- [5] Steffen Bleul, Thomas Weise, and Kurt Geihs from the University of Kassel, FB-16, Distributed Systems Group, Wilhelmshöher Allee 73, 34121 Kassel, Germany. Large-Scale Service Composition in Semantic Service Discovery. In *Ws-Challenge Part: M. Brian Blake, Andreas Wombacher, Michel C. Jaeger, and William K. Cheung, editors, Proceedings of 2006 IEEE Joint Conference on E-Commerce Technology (CEC'06) and Enterprise Computing, E-Commerce and E-Services (EEE'06)*, pages 427–429, 2006. In proceedings [18]. 1st place in 2006 WSC. Online available at <http://www.it-weise.de/documents/files/BWG2006WSC.pdf> [accessed 2008-10-21]. See 2007 WSC [6] and [7].
- [6] Steffen Bleul, Thomas Weise, and Kurt Geihs from the University of Kassel, FB-16, Distributed Systems Group, Wilhelmshöher Allee 73, 34121 Kassel, Germany. Making a Fast Semantic Service Composition System Faster. In *Proceedings of IEEE Joint Conference (CEC/EEE 2007) on E-Commerce Technology (9th CEC'07) and Enterprise Computing, E-Commerce and E-Services (4th EEE'07)*, pages 517–520, 2007. In proceedings [19]. 2nd place in 2007 WSC. Online available at <http://www.it-weise.de/documents/files/BWG2007WSC.pdf> [accessed 2008-10-21]. See 2006 WSC [5] and [7].
- [7] Thomas Weise, Steffen Bleul, and Kurt Geihs from the University of Kassel, FB-16, Distributed Systems Group, Wilhelmshöher Allee 73, 34121 Kassel, Germany. Web Service Composition Systems for the Web Service Challenge – A Detailed Review. *Kasseler Informatikschriften (KIS) 2007, 7*, University of Kassel, FB16, Distributed Systems Group, Wilhelmshöher Allee 73, 34121 Kassel, Germany, University of Kassel, November 19, 2007. Persistent Identifier: urn:nbn:de:hebis:34-2007111919638. Online available at <http://kobra.bibliothek.uni-kassel.de/handle/urn:nbn:de:hebis:34-2007111919638> and <http://www.it-weise.de/documents/files/WBG2007WSCb.pdf> [accessed 2007-11-20]. See also [5, 6].
- [8] *UDDI Version 2.04 API Specification*. OASIS, July 2002. <http://uddi.org/pubs/ProgrammersAPI-V2.04-Published-20020719.pdf> [accessed 2008-10-20].
- [9] Daniel Steinberg Stuart Cheshire. *Zero Configuration Networking: The Definitive Guide*. O'Reilly Media, Inc., 2005.
- [10] Stuart Cheshire and Marc Krochmal. DNS-Based Service Discovery. IETF, Internet-Draft, August 2006.

- [11] Stuart Cheshire and Marc Krochmal. Multicast DNS. IETF, Internet Draft, August 2006.
- [12] Yaron Y. Golland, Ting Cai, Paul Leach, and Ye Gu. Simple Service Discovery Protocol/1.0. IETF, Expired, April 2000.
- [13] E. Guttman, C. Perkins, J. Veizades, and M. Day. Service Location Protocol, Version 2. RFC 2608, June 1999.
- [14] Vincent Lenders, Martin May, and Bernhard Plattner. Service Discovery in Mobile Ad Hoc Networks: A Field Theoretic Approach. In *WOWMOM '05: Proceedings of the Sixth IEEE International Symposium on World of Wireless Mobile and Multimedia Networks*, pages 120–130, 2005. IEEE Computer Society, Washington, DC, USA. ISBN: 076-9-52342-001-. doi:10.1109/WOWMOM.2005.93.
- [15] Bill Segall, David Arnold, Julian Boot, Michael Henderson, and Ted Phelps. Content Based Routing with Elvin4. In *AUUG2K*, 2000.
- [16] Roland Reichle, Michael Wagner, Mohammad Ullah Khan, Kurt Geihs, Jorge Lorenzo, Massimo Valla, Cristina Fra, Nearchos Paspallis, and George A. Papadopoulos. A Comprehensive Context Modeling Framework for Pervasive Computing Systems. In *8th IFIP International Conference on Distributed Applications and Interoperable Systems (DAIS)*, June 2008. Springer Verlag, Oslo, Norway. to appear.
- [17] *Proceedings of IEEE Joint Conference (CEC/EEE 2008) on E-Commerce Technology (Tenth CEC'07) and Enterprise Computing, E-Commerce and E-Services (Fifth EEE'08)*, July 21–24, 2008, Washington, D.C., District of Columbia, USA. IEEE Computer Society, IEEE Computer Society, 10662 Los Vaqueros Circle, P.O. Box 3014, Los Alamitos, CA 90720-1314. ISBN: 978-0-76953-340-7. Library of Congress Control Number: 2005928254. BMS Part Number CFP08321-PRT.
- [18] *Proceedings of 2006 IEEE Joint Conference on E-Commerce Technology (CEC'06) and Enterprise Computing, E-Commerce and E-Services (EEE'06)*, June 26–29, 2006, The Westin San Francisco Airport, 1 Old Bayshore Highway, Millbrae, United States. IEEE Computer Society, Los Alamitos, California, Washington, Tokyo. ISBN: 978-0-76952-511-2.
- [19] *Proceedings of IEEE Joint Conference (CEC/EEE 2007) on E-Commerce Technology (9th CEC'07) and Enterprise Computing, E-Commerce and E-Services (4th EEE'07)*, July 23–26, 2007, National Center of Sciences, Tokyo, Japan. IEEE Computer Society, IEEE Computer Society. ISBN: 978-0-76952-913-4.

Citation Suggestion

```
@inproceedings{BWG2008GSDFMR,  
  title      = {Geminga: Service Discovery for Mobile Robotics},  
  author     = {Philipp Andreas Baer and Thomas Weise, and Kurt Geihns},  
  affiliation = {University of Kassel, FB-16, Distributed Systems Group,  
                Wilhelmsheer Allee 73, 34121 Kassel, Germany},  
  booktitle  = {Proceedings of The Third International Conference on  
                Systems and Networks Communications, ICSNC 2008},  
  month      = {oct # {~26--31},},  
  year       = {2008},  
  location   = {The Palace Hotel, Sliema, Malta},  
  publisher  = {IEEE Computer Society},  
  address    = {10662 Los Vaqueros Circle, Los Alamitos,  
                California 90720-1314, USA},  
  url        = {http://www.it-weise.de/documents/files/BWG2008GSDFMR.pdf},  
}
```