Linked Data Integration for Semantic Dialogue and Backend Access

Daniel Sonntag and Malte Kiesel
DFKI - German Research Center for AI
Saarbrücken and Kaiserslautern, Germany
daniel.sonntag@dfki.de, malte.kiesel@dfki.de

Abstract

A dialogue system for answering user questions in natural speech presents one of the main achievements of contemporary interaction-based AI technology. Modern dialogue frameworks function as middleware between the interface component and the backend where the answers to the user questions are stored in heterogeneous formats. We implemented an interface to linked data sources as part of a complex natural language understanding and semantic retrieval process, thereby integrating the querying and answering task into a common framework. The semantic backend system integrates multiple linked data sources to allow for an advanced multimodal question answering (QA) dialogue.

Introduction

Over the last several years, the market for speech technology has seen significant developments (Pieraccini and Huerta 2005) and powerful commercial off-the-shelf solutions for speech recognition (ASR) or speech synthesis (TTS). Further application scenarios, more diverse and dynamic information sources, and more complex prototype systems need to be addressed in the context of QA. Dialogue-based QA allows a user to pose questions in natural speech, followed by answers presented in a concise form (Sonntag et al. 2007). For example, "Who is the German Chancellor?" The short answer is "Angela Merkel."

We hypothesise that: *Whereas the dialogue-based access to keyword-based search engines has only moderate success, semantic (ontology-based) interpretations of dialogue utterances may become the key advancement in semantic search, thereby mediating and addressing dynamic semantic search engines which are already freely available.*

We consider the Linked Data Cloud (Bizer 2009) an example of as such an (abstract) semantic search engine. We think that the RDF triple structure itself, which is used in Linked Data, represents enough structure to be called a database index which maps a wildcard triple pattern onto the matching concrete data triples. The same idea has been followed by the semantic search engines Sindice.com and Sig.ma.

In our approach, we rely on declarative mappings between a given data source (in a particular structure, e.g., ontologies) to corresponding Web resources retrieved from Sindice, Sig.ma, or remote RDF repositories. Thereby we assume that the Web resources have an associated RDF description in order to comply with the requirements for Semantic Web data retrieval (Fensel et al. 2003). This requirement makes Linked Data, which advocates the use of URIs to uniquely identify things in the world, particularly interesting ((Sauermann and Cyganiak 2008). We also assume that useful information for the URIs is provided in RDF when resolving the URIs, so that we can exploit the triples behind a URL. The exciting possibilities lie in a holistic approach for automatic linked data discovery through the semantic search index (Tummarello, Oren, and Delbru 2007) and/or direct SPARQL queries, as well as the result presentation within a dialogue-based QA process which is the focus of this paper.

We learned some lessons which we use as guidelines in the development of multimodal dialogue systems where users can combine speech and gestures when using multiple interaction devices. In earlier projects (Wahlster 2003; Reithinger et al. 2005) we integrated different sub-components to multimodal interaction systems. Other lessons served as guidelines in the development of *semantic* dialogue systems (Oviatt 1999; Sonntag et al. 2007). These systems have four main properties: multimodality, the (ontological) representation of interaction structures and queries, a semantic representation of the interface, and encapsulation. In this contribution, we address the representation of query structures and the encapsulation aspect in particular. This is because a Linked Data integration for semantic dialogue and backend access mainly addresses these two issues. A third special issue exists in the context of Linked Data information sources, namely characteristics of trust/explanation. Hence, we will describe our approach toward establishing trust/explanation in dialogue-based linked data retrieval by following (Glass, McGuinness, and Wolverton 2008).

This paper is structured as follows. First, we will describe the dialogue system architecture, then focus on the backend retrieval step as a part of a typical (multimodal) interaction cycle. We will describe the Linked Data access and the corresponding semantic mediation (and trust/explanation) effort. Following that, we will provide an integrated dialogue example. The paper also provides a conclusion.

Dialogue System Architecture

The dialogue system architecture is based on a generic framework for implementing multimodal dialogue systems (ODP platform, available at http://www.semvox.de/). Conceptually, ODP allows us to implement the representation and encapsulation requirements.

Representation: In a complex interaction system, a common ground of terms and structures is absolutely necessary. A shared representation and a common knowledge base ease the dataflow within the system and avoid costly and error-prone transformation processes (c.f. “No presentation without representation”). More precisely, an ontology-based representation of a user query can be used to infer a SPARQL query that can be posed to a Linked Data endpoint.

Encapsulation: Encapsulate the dialogue interface proper from the application backend as much as possible. The application backend comprises of several information sources, and single Linked Data SPARQL endpoints represent individual information sources. This is the most important architectural commitment; multiple user interfaces can be used and the dialogue system acts as middleware between the multimodal interface and the backend access, i.e., several RDF repositories which include the Linked Data repositories.

Technically, the generic dialogue framework follows a programming model which eases the interface to external third-party components (e.g., the automatic speech recogniser (ASR), natural language understanding (NLU), or synthesis component). The client provides means to connect to the dialogue system via the *event bus*, to notify it of occurred events, to record and play back audio streams, and to render the received display data obtained from the *dialogue system/dialogue manager* (figure 1). In the context of semantic search, however, the interesting thing is that ODP uses ontology concepts in a model-based design. This means all internal and external module messages are based on ontology structures. The dialogue system contains an ontology-based rule engine for processing dialogue grammars and an external service connector (which implements the Java interface to the third-party components and the backend, see figure 2). The dialogue system acts as the *middleware* between the clients and the backend services (that hide complexity from the user by presenting aggregated ontological data or the results from the Linked Data lookup).

Backend Access

Figure 3 provides a rough sketch of the basic processing chain within the typical (multimodal) interaction cycle.

Answering natural language queries is a complex process. Apart from basic natural language analysis (ASR+NLU), several mapping, disambiguation, and (de)composition steps need to be executed. Let’s take a look at these steps: *1)* Natural language analysis (Engel 2006) finds named entities, verbs, etc., and constructs a structured representation of the query. This representation still builds on the natural language version of the query. *2)* A mapping of entity (labels) to identifiers (URIs) is also needed. *3)* A mapping of the structured representation of the query to a formal query

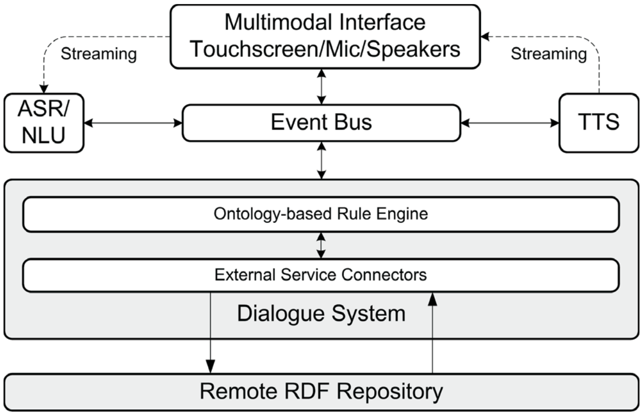

Figure 1: Dialogue system architecture

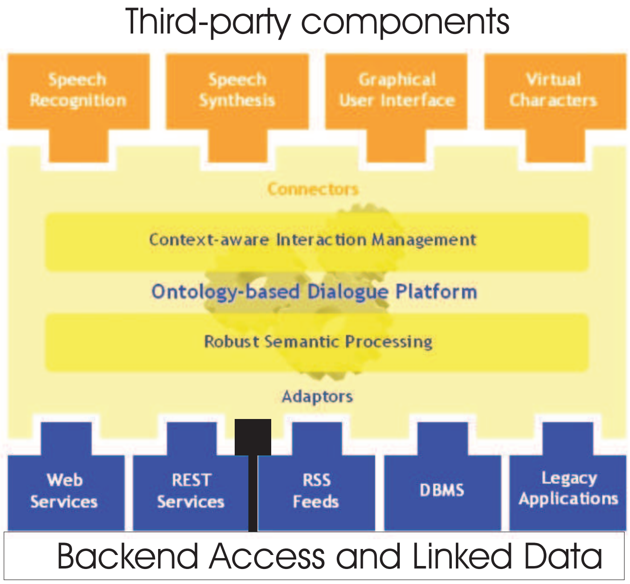

Figure 2: Backend access as new external ODP service connector

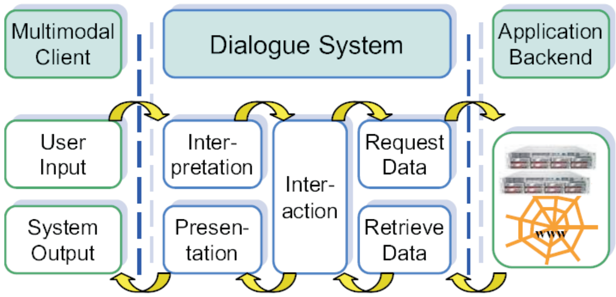

Figure 3: Typical (multimodal) interaction cycle

representation is done. The formal representation does not use concepts of verbs, nouns, etc., anymore, but rather uses a custom knowledge representation. If only one query answering backend exists, the knowledge representation of this backend can be used. Otherwise, an interim representation is generated. (We call these structures eTFS which stands for extended Typed Feature structures.) *4)* The formal query is analysed and mapped to one or more services that can answer (parts of) the query. This step typically involves several substeps including decompositing the query into smaller parts, finding suitable services (service discovery), mapping the query to other representations, planning a series of service executions, and possibly clarification requests or disambiguation requests. *5)* The query is sent to several backend services (in our case, a service composition module and a semantic mediator). *6)* Results (using the respective backend's knowledge representation) are mapped to the custom knowledge representation. *7)* If multiple results are found, results have to be merged. This may involve finding identical entities in individual results, and combining them. Without explicit mapping information, this is an error-prone process. Because of this, we try to mediate the results from different information sources by presenting results incrementally or, for example, in different regions of a large touchscreen-based presentation environment.

Some of these steps can be combined. For example, the entity mapping of the natural language analysis outputs to identifiers can be deferred until more of the actual query structure or its mapping to the knowledge backend is known. This allows us to include background knowledge for the disambiguation of user inputs and prevents the system from building formal queries that may not get answered by the backend systems. However, these kinds of optimisations are very difficult in case the system is not monolithic (i.e., multiple backends and knowledge representations are used).

Our system is quite a lightweight variant, implementing only some of the features outlined above. Basically, one backend is used for formalised queries. Additional services provide translation facilities and other simple transformation actions as well as supporting semantic keyword search preprocessing.

We use an RDF store (Jena TDB[1]) with the RDF version of Yago (**?**) which is a structured representation of Wikipedia contents. Another service can enrich answers provided by that backend with further information, based on Linked Data information sources. As an intermediate query representation language, a syntax based on the SPARQL Inferencing Notation (SPIN[2]) is used. This is in effect a structured version of SPARQL, the RDF query language. SPARQL originally uses a custom plain-text/string query syntax similar to SQL. The SPIN format instead uses an RDFS vocabulary to represent the individual SPARQL terms. The RDF(S) structured syntax allows us to use rule engines and other RDF(S)-based tools to modify the actual query. To make changes to the query is much more complicated in the original SPARQL string representation. The rule engine which is integrated into ODP and used to route and modify queries as needed also provides debugging tools and IDE support for writing rules.

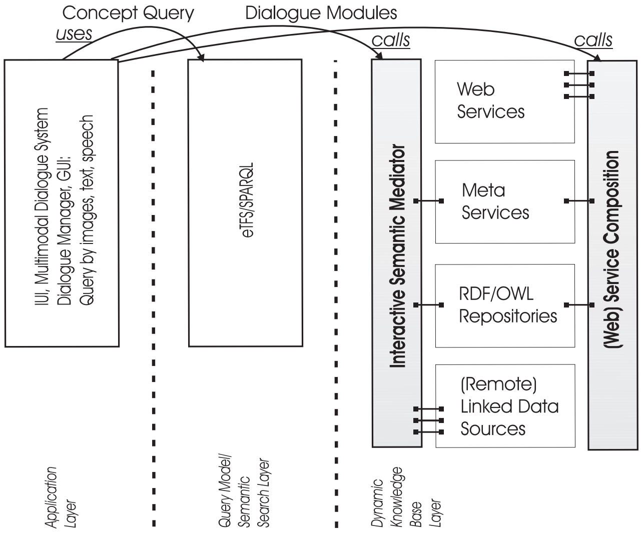

Figure 4: Three-tier semantic search architecture including the semantic mediator and the service composition module

The technical semantic search architecture comprises of three tiers: the application layer (user interface, dialogue system/manager), the query model/semantic search layer (eTFS/SPARQL structures), and the dynamic knowledge bases layer for the application backend (figure 4).

Let's have a look at a simple query for the songs by "Miley Cyrus" in order to better understand how the concept query (figure 5) is processed by the (Web) Service composition modules or interactive semantic mediator.

First, the NLU output representing this query can roughly be described in an ad-hoc syntax: `Given: Person("Miley", "Cyrus"), Query focus: "created"`. Figure 5 shows the query after its transformation to the eTFS/SPIN syntax (note the example shown here has been shortened a bit manually). Some mapping steps (i.e., mapping the term "Miley Cyrus" to the appropriate Yago resource, and resolving the predicate to ask for) have already been performed.

Second (performed by the composition module), this query is executed by a service calling a SPARQL endpoint. The answer is then transformed again into the eTFS syntax using a schema and variable bindings based on the SPARQL RDF results syntax.

Third (performed by the composition or mediator module), with the information found in the result, additional services can be triggered. Let us assume one registered service is able to find further information about entities of the type "Medium". "2006 singles" is a subclass of that type in the Yago type hierarchy. Therefore, the service is selected by the rule engine as a candidate to find more information about the first result (found with type "2006 singles").

Fourth (performed by the mediator module), additional Linked Data sources and meta services can be addressed.

[1] http://jena.sourceforge.net/TDB/

[2] http://spinrdf.org/

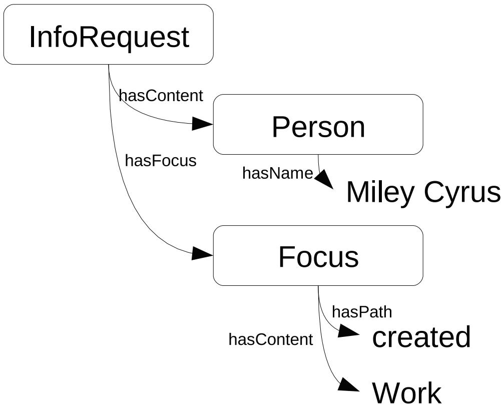

Figure 5: NLU representation (concept query)

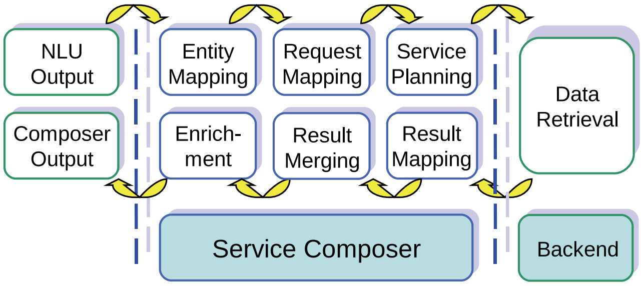

Figure 6: Service composer module

Service Composer Module

The service composer module (figure 6) takes input from the NLU module (step *1)* as explained. Entity and request mapping steps *2)* and *3)* are performed next, then a service execution plan is computed *4)* and *5)*, and the results are mapped, merged, and enriched with priority information *6)* and *7)*.

Semantic Mediator Module

Information integration is generally concerned with access to heterogeneous information sources to be mediated in order to provide an integrated view of the data. There is (in addition to the terminological) also a structural difference between the entities in the Linked Data cloud and the discourse ontology which we use. In this situation, very different semantic resources have to be matched on the query or answer side. The semantic mediator module (figure 7) should provide the necessary transformations especially between the structurally different data sources. Linked Data sources (normally at SPARQL endpoints) are addressed by the help of this component. Additionally, the use of Sindice allows us to formulate a simple but powerful backend service that aggregates search terms to URIs in indexed RDF documents. This functionality is often undervalued in the context of a complex information retrieval system.

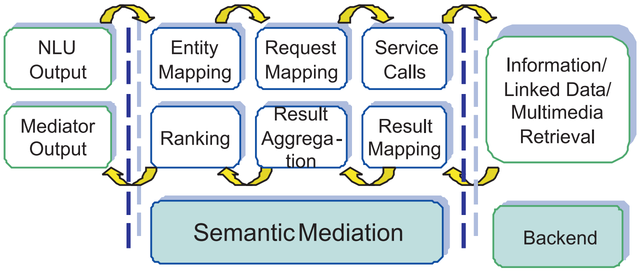

Figure 7: Semantic mediator module

Trust/Explanation: Within the semantic mediator, we do not directly address the answer merging problem (unlike the service composer) since the access to the multimodal interface allows for the incremental presentation of results from different information sources. In combination with user initiative dialogue feedback "This information stems from the Internet (... and is not entirely certain)", the user is not only presented the factual answers to questions, but also some explanations about the QA process. As (Glass, McGuinness, and Wolverton 2008) show, proper explanations are one main factor that influence the level of user trust in complex (adaptive) AI systems. Using natural dialogue for explanations in dialogue-based QA is a unique opportunity for interactive semantic mediation.

(Meta) Web Services

Some services found on the web and used by the system represent **meta services**. For example, Sindice.com is a search engine that can look up semantic web resources by label, providing a simple way of resolving entities and discovering linked data endpoints relevant to the query at hand. Also, we have been experimenting with services that provide keyword-to-SPARQL query functionality—for the basic idea, see (Duc Thanh Tran and Cimiano 2009). These services can be directly used as key components in the service composition process. We expect similar meta services to appear soon, for example services analysing SPARQL queries and recommending other relevant data sources or rewriting/federating the queries based on the *voiD* vocabulary (Alexander et al. 2009). This would move much of the effort required to aggregate and query the Web of Data from the user side to the server side.

Apart from the services which help the composition process, there are also a number of **standard web services**[3] such as YouTube; we mapped the GAPI results onto a ranked result table containing several videos with metadata (title, list of comments, etc.). The ranking is done by computing a string-based distance between the (keyword) query terms extracted from the eTFS representation and the titles of the first and second result pages.

Remote Linked Data Sources

The structure of the knowledge layer conveys the data timeliness of the Linked Data and the corresponding semantic search engines. Accordingly, the backend access foresees

[3] By *web service*, we not only consider SOAP services, but also services based on REST, JSON, XML-RPC, and so on.

the direct access of these online sources (via Web API connectors or SPARQL endpoint connectors), instead of maintaining a local RDF store where the other internal data sources for dialogue processing and multimodal dialogue remain (cf. the three-tier semantic search architecture in figure 4). The Linked Data Access delivers the following additional multimedia information about Miley Cirus (based on an appropriate mapping, the access to further music-related Linked Data sources is easily conceivable).

- DBpedia:

```
associatedBand:Jonas Brothers
background:''solo\_singer''
birthdate:1992-11-23
genre:Pop rock
```

- Jamendo: is a community collection of music all freely licensed under Creative Commons licenses. This data retrieval step provides information about image, home page, and the (home) location of Miley.
- MusicBrainz: provides data about Miley and albums, e.g., names, release titles, and track lists.

Integrated Dialogue Example

We will illustrate how Linked Data are integrated in a semantic/ontology-based QA dialogue and backend access. This demonstrates a new functionality for complex QA dialogue systems, where specific information about artists and multimedia material can be retrieved from backend sources including Linked Data sources, as well as being enriched by additional results. According to the following search pattern instantiation (this representation is based on the SPARQL Inferencing Notation SPIN), QA-based dialogue can be conducted. The following XML snippet represents a (shortened) query as it occurs in our system:

```
<object type="sp#Select">
  <slot name="sp#where">
    <object type="sp#Triple">
      <slot name="sp#subject">
        <object type="sp#Resource">
          <slot name="sp#hasUri">
            yago#Miley_Cyrus
          </slot>
        </object>
      </slot>
      <slot name="sp#predicate">
        <object type="sp#Resource">
          <slot name="sp#hasUri">
            yago#created</value>
          </slot>
        </object>
      </slot>
      <slot name="sp#object">
        <object type="sp#Variable">
          <slot name="sp#varName">
            <string>work</string>
          </slot> ...
```

Note that domain specific URIs are partially represented as literals—this is due to the fact that the query is generated from an internal representation using lightweight ontologies instead of the complete domain ontologies. In the TFS/ODP framework, unresolved references (with pointers to domain ontologies and facts) are not supported.

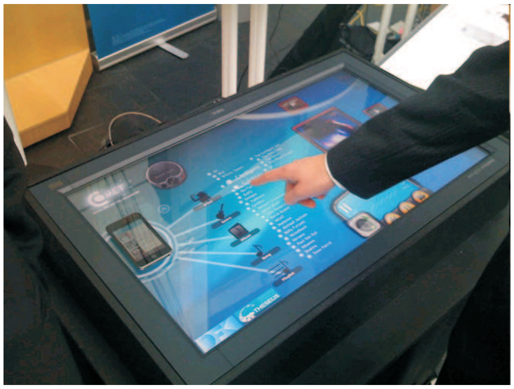

Figure 8: Multitouch user interface, see (Sonntag, Deru, and Bergweiler 2009).

Results are represented in a similar fashion, using a TFS variant of the SPARQL result RDF syntax. In these results, links to domain ontologies are preserved and can be used in further dialogue actions. The natural interaction character of dialogue systems is thereby applied to Linked Data sources.

The ODP rule engine may be used to modify, enrich, and contextualise all the data contained in the TFS representations. Currently, this feature is used for several transformation steps but also for enriching primary results with additional information fetched from Linked Data sources.

Further interaction possibilities are shown in the following example dialogue. The dialogue described the power of speech-based interaction in the context of one of our demonstrator systems (Comet, see (Sonntag, Deru, and Bergweiler 2009)) which aims at the design and implementation of combined mobile and touchscreen-based multimodal Web 3.0 interfaces. The dialogue example shows the meta web service access to YouTube via the GAPI; figure 8 shows the multitouch user interface of the multimodal dialogue system at the application layer. Currently, the integration of Linked Data in the system is limited to providing additional complementary information only, but we will work on the extension of the role of Linked Data in the future.

1. **U:** The user reads a semantic web page about musicians and puts, e.g., by a pointing gesture on the screen, Miley Cirus in focus.
2. **U:** “Which songs are from her?”
3. **S:** “I will search for song information about the artist: Miley Cirus”
4. **S:** (“I will search the Internet for a suitable answer.”)
5. **S: “I found some singles from 2006.”** (+ display of a short supportive text in the media screen)
6. **S: “There are some videos available from YouTube.”** (+ display of a media screen)

Conclusion

We described a complex AI system architecture for dialogue-based question answering. In the context of ontology-based dialogue systems, we discussed some of the main issues, i.e., "representation" and "encapsulation". A comprehensive overview of ontology-based dialogue processing and the systematic realisation of these properties can be found in (Sonntag 2010), pp.71-131. "Trust/explanation" is a further issue, which comes into account when addressing Linked Data sources. Our dialogue-based solution is to inform the user about the information sources (or easy-to understand surrogates such as "the Internet") during the dialogue. In addition, heterogeneous *textual* answer snippets or multimedia material from Linked Data sources can be displayed as an additional information on a larger presentation screen, e.g., a touchscreen. Our semantic backend system integrates multiple linked data sources to allow for an advanced multimodal QA dialogue thereby combining service composition with general semantic mediation of heterogeneous information sources. This allows us to combine data and knowledge retrieval with information and multimedia retrieval. Our impression is that data stemming from Linked Data sources should be treated as stemming from a new answer stream which we named *Remote Linked Data Source*. Future dialogue systems build upon the multimodal speech-based discourse and dialogue infrastructure. A first industrial dissemination of the Linked Data access for a radiologist is planned (Sonntag, Wennerberg, and Zillner 2010).

Acknowledgments

This research has been supported by the THESEUS Programme funded by the German Federal Ministry of Economics and Technology (01MQ07016). Thanks go out to Robert Nesselrath, Yajing Zang, Matthieu Deru, Simon Bergweiler, Gerhard Sonnenberg, Norbert Reithinger, Gerd Herzog, Alassane Ndiaye, Tilman Becker, Norbert Pfleger, Alexander Pfalzgraf, Jan Schehl, Jochen Steigner, and Colette Weihrauch for the implementation and evaluation of the dialogue infrastructure.

References

Bizer, C. 2009. The emerging web of linked data. *IEEE Intelligent Systems* 24(5):87–92.

Engel, R. 2006. SPIN: A Semantic Parser for Spoken Dialog Systems. In *Proceedings of the 5th Slovenian and First International Language Technology Conference (IS-LTC 2006)*.

Fensel, D.; Hendler, J. A.; Lieberman, H.; and Wahlster, W., eds. 2003. *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential*. MIT Press.

Glass, A.; McGuinness, D. L.; and Wolverton, M. 2008. Toward establishing trust in adaptive agents. In *IUI '08: Proceedings of the 13th international conference on Intelligent user interfaces*, 227–236. New York, NY, USA: ACM.

Oviatt, S. 1999. Ten myths of multimodal interaction. *Communications of the ACM* 42(11):74–81.

Pieraccini, R., and Huerta, J. 2005. Where do we go from here? research and commercial spoken dialog systems. In *Proceeedings of the 6th SIGDdial Worlshop on Discourse and Dialogue*, 1–10.

Reithinger, N.; Fedeler, D.; Kumar, A.; Lauer, C.; Pecourt, E.; and Romary, L. 2005. MIAMM - A Multimodal Dialogue System Using Haptics. In van Kuppevelt, J.; Dybkjaer, L.; and Bernsen, N. O., eds., *Advances in Natural Multimodal Dialogue Systems*. Springer.

Sonntag, D.; Engel, R.; Herzog, G.; Pfalzgraf, A.; Pfleger, N.; Romanelli, M.; and Reithinger, N. 2007. *SmartWeb Handheld—Multimodal Interaction with Ontological Knowledge Bases and Semantic Web Services*, volume 4451 of *Lecture Notes in Computer Science*. Springer. 272–295.

Sonntag, D.; Deru, M.; and Bergweiler, S. 2009. Design and Implementation of Combined Mobile and Touchscreen-Based Multimodal Web 3.0 Interfaces. In *Proceedings of the International Conference on Artificial Intelligence (ICAI)*.

Sonntag, D.; Wennerberg, P.; and Zillner, S. 2010. Applications of an ontology engineering methodology accessing linked data for medical image retrieval. *AAAI Spring Symposium on Linked Data Meets Artificial Intelligence*.

Sonntag, D. 2010. *Ontologies and Adaptivity in Dialogue for Question Answering*. AKA and IOS Press, Heidelberg.

Tummarello, G.; Oren, E.; and Delbru, R. 2007. Sindice.com: Weaving the open linked data. In Aberer, K.; Choi, K.-S.; Noy, N.; Allemang, D.; Lee, K.-I.; Nixon, L. J. B.; Golbeck, J.; Mika, P.; Maynard, D.; Schreiber, G.; and Cudr-Mauroux, P., eds., *Proceedings of the 6th International Semantic Web Conference and 2nd Asian Semantic Web Conference (ISWC/ASWC2007), Busan, South Korea*, volume 4825 of *LNCS*, 547–560. Berlin, Heidelberg: Springer Verlag.

Wahlster, W. 2003. SmartKom: Symmetric Multimodality in an Adaptive and Reusable Dialogue Shell. In Krahl, R., and Günther, D., eds., *Proceedings of the Human Computer Interaction Status Conference 2003*, 47–62. Berlin, Germany: DLR.