Language-Derived Information and Context Models

Andre Blessing*, Stefan Klatt*, Daniela Nicklas†, Steffen Volz‡ and Hinrich Schütze*

*Institute for Natural Language Processing
University of Stuttgart
Azenbergstr. 12
D-70174 Stuttgart, Germany

†Applications of Parallel and Distributed Systems
University of Stuttgart
Universitätsstr. 38
D-70569 Stuttgart, Germany

‡Institute for Photogrammetry
University of Stuttgart
Geschwister-Scholl-Str. 24D
D-70174 Stuttgart, Germany

Abstract

There are a number of possible sources for information about the environment when creating or updating a context model, including sensorial input, databases, and explicit modeling by the system designer. Another source is natural language, either in the form of electronic text (e.g., the world wide web) or speech. In this paper, we investigate the implications for context models when some of their information is derived linguistically with an emphasis on the issues of hybrid models and mapping between entities in language and context model. We present a prototype that tests some of our ideas.

1. Introduction

Context-aware applications require knowledge about the context of their users and must adapt their behavior according to that knowledge. Since modeling and managing context information is tedious and expensive, it is beneficial to share context models between applications as already implemented by various platforms (e.g. [11, 10, 4, 3]).

Context models are intended to mirror the state of the environment as closely as possible. This raises the challenge of how to update the context model when the environment changes. Many changes are amenable to detection by sensors, e.g., temperature and lighting conditions. This information is often referred to as lower context: context-aware applications have to interpret these raw values to derive the information they are interested in (situations or higher context). For example, a GPS sensor gives the coordinates of a person, which have to be compared with a map to find out whether the person is in a certain location, e.g. a shopping mall (which would be higher context).

Other changes are 'situational' rather than corresponding to a simple perceivable datum. For example, the conversion of a two-way street into a one-way street is hard to detect by the interpretation of sensor data. Sensors would, e.g., only reveal that the movement profiles of cars passing this street have changed, but the situation that results from the change (namely the conversion) cannot be unambiguously identified. Thus, such semantic changes of the real world require a manual and costly update of context models. To advance the state of the art and to allow for a higher degree of automation with respect to the update of semantic changes within context models, further information sources have to be considered.

One such possible source is natural language, either in the form of electronic text (e.g., the world wide web or text entered by a user) or speech. Since language is the main medium of communication between people and since content communicated via language is typically higher context, a large amount of contextual information is most accessible or can be most easily elicited in language. For example, newspaper stories alert readers to planned road constructions that will slow down traffic.

We suggest that information derived from human language (text or speech) using Natural Language Processing (NLP) can close the gap between immediate sensorial updates and long-term manual updates of context models via geographic databases – which will eventually catch up with the state of the world and include one-way streets and similar recent changes. The challenge is to efficiently extract this unstructured information in a form that can be used by context-aware applications for updating context models.

In this paper, we investigate the implications for context models when some of their information is derived linguistically. Most of these implications, e.g. the representation of uncertainty, are important in general for robust and accurate modeling, but of more immediate concern here.

In the next section, we will briefly cover related work. Then we discuss how to get from text to context information and a number of reasoning and modeling issues. We introduce the general architecture of a text sensor and show

how our prototype, the Traffic Analyzer, implements it partially. We conclude with some remarks on future research.

2 Related Work

There are a number of systems that use context models to support NLP applications, but to our knowledge NLP techniques have not been used for updating context models.

A context-aware chat system that integrates textual information is presented in [9]. It is capable of contextual interpretation of terms depending on the context of the person writing them. For example, dollar amounts will be interpreted correctly depending on location (USA, Canada etc). An important distinguishing feature of our system is that the contents of the context model are partially derived from text (in addition to interpreting text based on the context model).

In [6], a context model is also used to interpret language correctly in a system that controls devices in an intelligent environment using an NLP interface. Here the context model assists the language understandig component. The authors show that the recognition error for speech can be reduced by 41% if context is used.

3 From Text to Context Information

Language is the most widely used medium for the encoding and communication of information. Unlike sensorial data, linguistic information is expressed at a high level of abstraction and possesses explanation facilities for representing data in a user-friendly way.

We now review three challenges that context models face when part of their informational content is derived from language: mapping of entities from language to the context model, representation of uncertainty, and hybrid models that are necessary for a heterogeneous representation.

3.1 Mapping

A sensor is usually tied to a particular entity in the context model. For a GPS sensor in a car, no further interpretation is required to map the position returned by the sensor to the car in the context model. For language input, linking entities in text with entities in the context model requires some form of inference. This inference is simple if the two entities happen to be described by the same string. Typical examples are street and city names. More problematic are cases of orthographic variation (e.g., München vs. Munich), misspellings (Sidney/Sydney) and abbreviations (Portola Drive vs. Portola Dr.). Mapping problems of this type can be solved by using NLP components like lexicons with a morphology component and different string matching methods.

The mapping task is particularly difficult if a geographic location has a more abstract designator like "end of highway," "border between France and Germany" or "the Bay Area". The context model probably does not contain these entities in this form. To resolve this type of reference some kind of ontology-based inference is necessary.

In summary, for language-derived information, in contrast to most sensors, some kind of inference mechanism is required to map entities in language to entities in the context model.

3.2 Representation of Uncertainty

The representation of uncertainty in context models is of general importance. Many aspects of context models, e.g., the GPS coordinates returned by a sensor, are associated with a certain margin of error. A different type of uncertainty is prevalent in natural language: ambiguity. Many words are ambiguous, that is, they have more than one meaning. For example, there are many Springfields in the United States. A second type of ambiguity concerns geographic names that also have a non-geographic referent, e.g., "Orlando" (also a common first name) and "Cambridge" (a university as well as a city). For any given occurrence of an ambiguous term, the correct referent has to be determined. Linguistic and non-linguistic context is needed to perform this selection. However, so-called disambiguation algorithms have a certain error rate. Therefore, some of the information extracted by an information extraction (IE) component will be uncertain. This uncertainty should be represented in the context model.

Uncertainty can be compounded if there are several sources that contradict each other. For example, two radio stations may broadcast conflicting reports about a traffic accident and the severity of resulting traffic problems.

As a consequence, a system with an NLP component should have some way of representing uncertainty. For context-aware systems, several approaches to uncertainty have been proposed. Ranganathan et al. describe a system that stores a certainty assessment with each unit of information in the context model [8]. Their system can accommodate different formalisms (e.g., Bayesian networks) to represent these certainty assessments. The Nexus Augmented World Model uses optional meta data tags for this [5].

3.3 Hybrid Models

There are different ways to represent a spatial context model. Two-dimensional and three-dimensional models represent each entity as a point on the plane or in 3D using geometric coordinates. Another type of model is the topological model, which is made up of edges and nodes. For example, a route planner represents intersections and streets as nodes and edges of a network, respectively. Geographic objects can also be represented ontologically as names and the relationships that hold between them. This symbolic representation is important for NLP applications to enable the linking of linguistic and geographic entities and relationships. A hybrid model accommodates several of these representations and appropriately links them with each other. In our system design and experiments, we have found that we need a hybrid model with all three representations for "NLP-enabled" context models. This is partly motivated by the requirements of the traffic application we will introduce below. But more importantly, the NLP component also needs a rich model for its IE component and for performing tasks like mapping. This distinguishes it from regular sensors, which do not need access to the contents of the context model. A temperature sensor can feed temperature into the context model without any knowledge about the contents of the model.

The need for hybrid models will be discussed below in more detail, but as an example consider the case of ambiguous place names mentioned earlier. Clearly, the knowledge captured in the context model will be useful when disambiguating Springfield in "Springfield on I91". In this case, a combination of a topological and an ontological model supports the desired inference. We believe that NLP "sensors" need access to the context model to extract complete and reliable information from natural language. All three levels of representation, geometric, topological and ontological, are necessary for this purpose.

4 Implementation

To evaluate our ideas, we implemented a text sensor that detects traffic jams in web news feeds. It uses context information from a context model to relate the extracted information to existing data and updates the context model with new information. For visualization, the application Traffic Analyzer displays this context information on a map.

4.1 Architecture

Figure 1 shows a conceptual model of a text sensor. It queries a context model for recent context data in order to relate the analyzed information with the existing context model. The Traffic Analyzer needs a geographic model of highways and exits as a context model; a text sensor for parties and other social events would need venue locations.

Note that in our case, this context model is the Augmented World Model which is managed by the Nexus platform, an open federation platform for context information that can be shared by many different applications. The Nexus platform is described in more detail in [7] and [3].

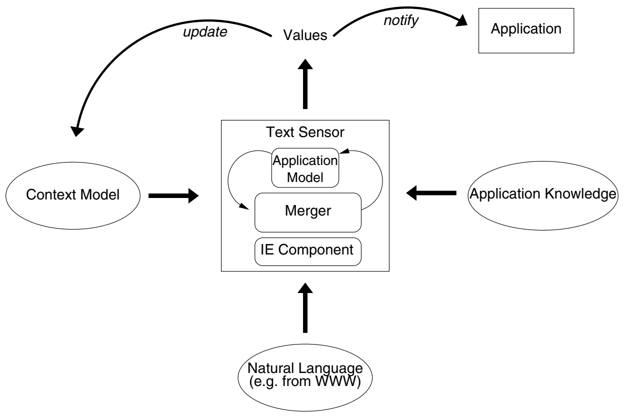


Figure 1. Architecture

In addition to the context model, the text sensor needs application knowledge that depends on its specific task and the information source it analyzes. For the domain of traffic analysis, these are rules like "traffic jams are reported between named exits" and "the order of the exits is relevant". This knowledge is used to transform the context model data to the so called application model that contains the information that is needed to guide the analysis process.

We have not formalized the application knowledge so far. In future work, we intend to build a domain specific ontology to automate the process of text sensor implementation.

An IE component analyzes the text from the natural language source and fills a template with the relevant information. Next, the Merger combines the information from the application model with the text to find valuable context data (in our example, traffic jams as geographical objects). These data can now be used to update the context model or to notify applications that are interested in such data.

4.2 Implementation of the hybrid model

We use a hybrid context model for our application. Figure 2 describes the different tiers of the model. The geometric tier is necessary for drawing maps. The topological tier allows us to check the consistency of traffic messages. Moreover, it is used to filter out possible candidates in the matching task.

The topological level represents vertical and horizontal relations. In the vertical case, there are two main relations. The first one is the inheritance relationship familiar from taxonomies with its advantages of representational economy and easier maintainability. In our case, we have a simple object with geo-spatial information as root. All other elements inherit from this object.

The second type of vertical relationship is containment or the part-of relationship (a city is part of a country, a street is part of a city etc.). Containment is the basic inference we need when reasoning about geographic entities. For example, in our application containment allows us to disambiguate the name of an exit by choosing the one that is contained in the global road segment that a traffic report is referring to.

There are also horizontal relations between entities at the same level, for example, the partial order of exits on a highway. Our application requires this part of the model because it is not always possible to infer the order from the coordinates of the exits. For instance, exits of different highways can be in close spatial proximity, so that location information alone is insufficient to identify an exit correctly from its linguistic description.

Finally, the symbolic tier is used for linking the entities from the information extraction component to the geographic data in the hybrid model.

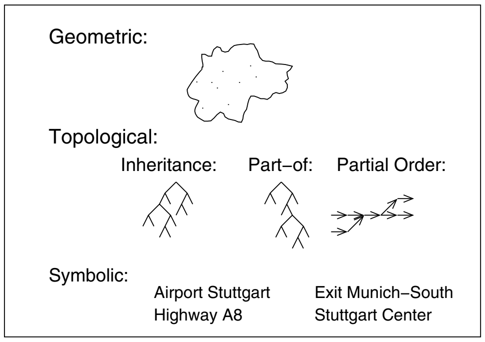
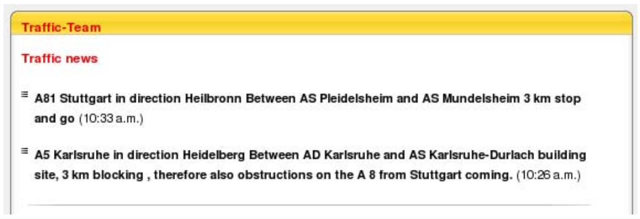


Figure 2. Application Context Model

4.3 The Information Extraction Component

Figure 3 shows an excerpt of a traffic report on the web site of the German broadcast station Antenne1[1].


Traffic-Team

Traffic news

- A81 Stuttgart in direction Heilbronn Between AS Pleidelsheim and AS Mundelsheim 3 km stop and go (10:33 a.m.)
- A5 Karlsruhe in direction Heidelberg Between AD Karlsruhe and AS Karlsruhe-Durlach building site, 3 km blocking , therefore also obstructions on the A 8 from Stuttgart coming. (10:26 a.m.)

Figure 3. Excerpts of a Traffic Report

Most of the messages have a relatively rigid format. They usually start with a number code for a highway (e.g. *A81*) or other street types. Next, global location points of the street are mentioned (e.g. city names like *Stuttgart* and *Heilbronn*), which can be more than a hundred miles away from each other, followed by local location points with a much smaller distance from 0-20 kilometers (e.g. *AS Pleidelsheim* and *AS Mundelsheim*; AS = exit).

[1]We translated the relevant terms from German to English.

Additional information about the traffic problem (e.g. obstructions, route diversions etc.) follow the description of the location. These additions are challenging for information extraction due to the large number of possible syntactic constructions that are used to express them.

The IE component first transforms the HTML format into plain text and then annotates the relevant information units in one pass by the application of a finite state transducer [12]. Figure 4 shows the annotated messages of the example report.

```
<m> <ROAD>A81</ROAD> <FROM>Stuttgart</FROM> in direction <TO>Heilbronn</TO>
Between <DFROM>AS Pleidelsheim</DFROM> and <DTO>AS Mundelsheim</DTO>
<LENGTH>3 km</LENGTH> <O>stop and go</O> (<RTIME>10:33</RTIME> a.m.) </m>

<m> <ROAD>A5</ROAD> <FROM>Karlsruhe</FROM> in direction <TO>Heidelberg</TO>
Between <DFROM>AD Karlsruhe</DFROM> and <DTO>AS Karlsruhe-Durlach</DTO>
<REASON>building site</REASON>, <LENGTH>3 km</LENGTH> <O>blocking</O> ,
<IMPL>therefore</IMPL> also <O>obstructions</O> on the <ROAD>A 8</ROAD>
from <DFROM>Stuttgart</DFROM> coming. (<RTIME>10:26</RTIME> a.m.) </m>
```

Figure 4. Annotated messages

Figure 5 shows the filled templates that were created by the template generator component from the annotated messages given in Figure 4. The id field of the template keeps track of how many different events were described by a single report. Note that the word *therefore* in the second message functions as a discourse marker that introduces a new event in the same message (the new event is encoded by a special incremented counter). The template is then passed on to the merger.

```
Message1:                          Message2:
id:     14.05.2005_11:00_1         id:     14.05.2005_11:00_2
status: IN                         status: IN
rtime:  10:33                      rtime:  10:26
road:   #1 A81                     road:   #1 A5 #2 A8
from:   #1 Stuttgart               from:   #1 Karlsruhe
to:     #1 Heilbronn               to:     #1 Heidelberg
dfrom:  #1 AS Pleidelsheim         dfrom:  #1 AD Karlsruhe #2 Stuttgart
dto:    #1 AS Mundelsheim          dto:    #1 AS Karlsruhe-Durlach
obs:    #1 stop and go             obs:    #1 blocking #2 obstructions
reason:                            reason: #1 building site
length: #1 3 km                    length: #1 3 km
```

Figure 5. Template fillings

4.4 Merging

The IE component only recognizes strings without establishing any relationship with entities in the context model. This is the job of the merger. Figure 6 shows the three steps in mapping between the filled template and our application model.

First the *road* name is selected and looked up in the symbolic tier. Our topological model represents each road as two ordered lists of the cities that are located on the road, corresponding to the two different directions (e.g., North-South and South-North). Depending on the direction indicated in the template (based on the slots *from* and *to*), the

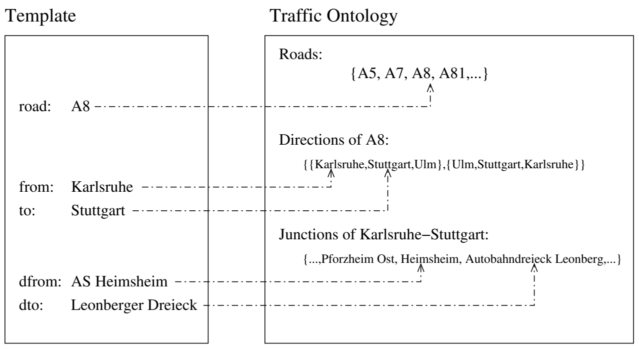
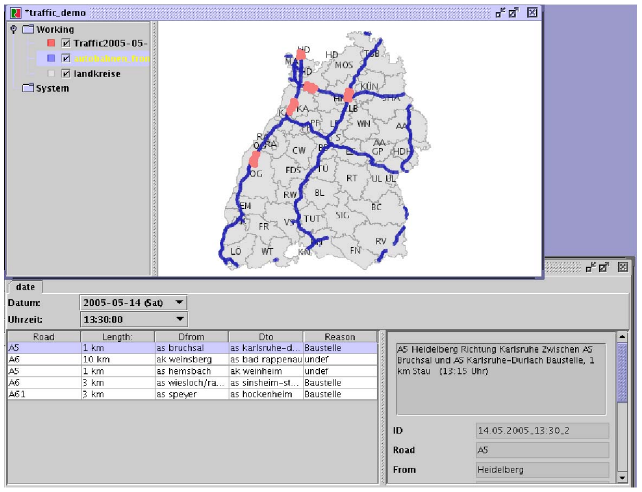

Template
Traffic Ontology
road: A8
from: Karlsruhe
to: Stuttgart
dfrom: AS Heimsheim
dto: Leonberger Dreieck
Roads:
{A5, A7, A8, A81,...}
Directions of A8:
{{Karlsruhe,Stuttgart,Ulm},{Ulm,Stuttgart,Karlsruhe}}
Junctions of Karlsruhe–Stuttgart:
{...,Pforzheim Ost, Heimsheim, Autobahndreieck Leonberg,...}

Figure 6. Mapping between template data and application model



Datum: 2005-05-14 (Sat)
Uhrzeit: 13:30:00

Road	Length:	Dfrom	Dto	Reason
A5	1 km	as bruchsal	as karlsruhe-d...	Baustelle
A6	10 km	ak weinsberg	as bad rappenau	undef
A5	1 km	as hemsbach	ak weinheim	undef
A6	3 km	as wiesloch/ra...	as sinsheim-st...	Baustelle
A61	3 km	as speyer	as hockenheim	Baustelle

A5 Heidelberg Richtung Karlsruhe Zwischen AS Bruchsal und AS Karlsruhe-Durlach Baustelle, 1 km Stau (13:15 Uhr)

ID 14.05.2005_13:30_2
Road A5
From Heidelberg

Figure 7. Screenshot of the prototype

relevant direction is selected. Finally, the segment of the road affected is identified based on the *dfrom* and *dto* tags.

The creation of this topological model of the area of interest is somewhat involved. Construction of some roads may not have been completed, so there can be gaps between exits. Another problem is the border of the area covered by our data. A road may leave the covered area and then reenter it at a different point. Furthermore, there are errors in the data set (e.g. incomplete streets) that need to be corrected.

Entities are mapped from the template to the context model by approximate string matching algorithms. Exact matches would not be sufficient because of misspellings, abbreviations and variations in how geographic objects are referred to, e.g. *Autobahndreieck Leonberg* vs. *Leonberger Dreieck* ("interchange triangle") in Figure 6.

4.5 Traffic Analyzer, a simple context-aware application

To test the use of language-derived information in context models, we implemented a small application with a graphical user interface (Figure 7). The basic functionality queries the current traffic situation. The application also allows the analysis of traffic patterns over time and subject to various constraints, for example, type of problem. There are also commercial systems (e.g. [2, 1]) for processing and visualizing traffic information, but these systems are custom-built for a particular application. They do not have a generic context-model-based architecture.

4.6 Evaluation

4.6.1 Evaluation of the IE component

We evaluated the information extraction (IE) component as well as the mapping component. For the evaluation of the IE component we analyzed traffic messages concerning highways of the German broadcast station *Antenne1* in a time period of four weeks that were updated every 30 minutes. As an evaluation set, we selected the 157 messages from the corpus with a time stamp of 12pm (noon). Table 1 shows the results for the extraction of the primary relevant geographical data that are important for the mapping process of the visualization component (see the discussion in the previous section[2]).

Corr, miss, and spur are the number of correct, missing, and spurious assignments, respectively. We use recall (rec) and precision (prec) for evaluation, defined as follows: rec=$\frac{corr*100\%}{corr+miss}$ and prec=$\frac{corr*100\%}{corr+spur}$.

	corr	spur	miss	rec	prec
ROAD	156	1	1	99.36	99.36
FROM	154	0	3	98.09	100.00
TO	154	0	3	98.09	100.00
DFROM+POS	132+16	0	2+7	94.27	100.00
DTO+POS	131+16	0+1	3+7	93.63	99.32

Table 1. Position extraction evaluation

The achieved accuracy rates are very promising. The errors for the tags `FROM`, `TO`, `DFROM`, `DTO` and `POS` occurred because our set of triggers was incomplete in one case. A fundamental issue with our current implementation is that the IE component cannot handle cases like the message in (1). Understanding this message requires the localization of an obstruction that is not explicitly mentioned in the message. A reasoning component is needed for more complex

[2]If a message is not marked with the tags *DFROM* and *DTO*, the first *POS*-tagged sequence was chosen as slot-filler element for the template slots *dfrom* and *dto*.

cases like this one.

(1)
```
<m> <POS>Interchange triangle
Karlsruhe</POS>, transition from
<ROAD>A8</ROAD> to <ROAD>A5</ROAD>
in direction <TO>Basel</TO> </m>
```

4.6.2 Evaluation of the mapping component

To evaluate the mapping component, we checked how well the different strings from the *road*, *dfrom* and *dto* slots were linked to geographic entities in the model.

An exact match occurred in only 5% of all cases. The main reason is the large number of abbreviations used in traffic messages. However, the string similarity matching algorithm was able to get the right match in 78% of cases. The remaining 17% were incorrect. Errors were mostly due to lack of coverage. The radio station covers traffic news for the state of Baden-Württemberg and surrounding areas, but our geographic data set only covers the state itself. Another reason is that some types of geographic entites are not covered in our data set, e.g., border crossings, service areas and tunnels. Since there will always be gaps in coverage we are planning to use evidence from the web in future versions of the system.

In this version of the prototype, uncertainty is only handled in the matching component. If similarity does not exceed a predefined threshold, then no match is returned. Extending uncertainty handling to more interesting cases is an important part of the future work we are planning.

5 Conclusion and Future Work

In this paper we have shown that by using NLP methods, we can tap new resources of context information and make it processable by context aware applications. We see a particularly high potential for text sensors detecting higher level context information that can be found in human-readable text documents, e.g. on the World Wide Web. However, to achieve high quality recognition, we need to customize text sensors with application knowledge and a basis of background information that can be obtained from context models.

We intend to replace our customized IE component with a generic parser that extracts all possibly relevant relationships. A customized component would then filter out those that are needed for the particular task. This approach would permit a clearer modularization of generic natural language parsing and specific geographic domain knowledge. The difficulty is to make linguistic knowledge about geography (e.g., the names of entities) available to the parser. Most generic parsers are not designed for this type of extensibility.

A more principled approach to uncertainty is proposed by [8] as discussed earlier. We plan to adopt a version of this approach for the representation of ambiguity and for indicating the reliability of information sources in cases where contradictory information is fed into the context model.

Acknowledgment

The Nexus project (Center of Excellence/SFB 627) is supported by the German Research Association (DFG).

References

[1] GEWI GmbH.

[2] OneStepAhead AG.

[3] M. Großmann, M. Bauer, N. Hönle, U.-P. Käppeler, D. Nicklas, and T. Schwarz. Efficiently managing context information for large-scale scenarios. In *Proceedings of the 3rd IEEE International Conference on Pervasive Computing and Communications*. IEEE Computer Society, 2005.

[4] K. Henricksen and J. Indulska. A software engineering framework for context-aware pervasive computing. In *Proc. of the 2nd IEEE International Conf. on Pervasive Computing and Communications*, 2004.

[5] N. Hönle, U.-P. Käppeler, D. Nicklas, and T. Schwarz. Benefits of integrating meta data into a context model. In *Proc. of 2nd IEEE PerCom Workshop on Context Modeling and Reasoning (CoMoRea)*. IEEE, 2005.

[6] L. H. Leong, S. Kobayashi, N. Koshizuka, and K. Sakamura. Casis: a context-aware speech interface system. In *IUI '05: Proceedings of the 10th international conference on Intelligent user interfaces*. ACM Press, 2005.

[7] D. Nicklas, M. Großmann, T. Schwarz, S. Volz, and B. Mitschang. A model-based, open architecture for mobile, spatially aware applications. In *Proceedings of the 7th International Symposium on Spatial and Temporal Databases: SSTD 2001; Redondo Beach, CA, USA*, July 2001.

[8] A. Ranganathan, J. Al-Muhtadi, and R. Campbell. Reasoning about uncertain contexts in pervasive computing environments. *IEEE Pervasive Computing*, 2004.

[9] A. Ranganathan, R. H. Campbell, A. Ravi, and A. Mahajan. Conchat: A context-aware chat program. *IEEE Pervasive Computing*, 2002.

[10] M. Roman and R. H. Campbell. Gaia: Enabling active spaces. In *Proceedings of the 9th ACM SIGOPS European Workshop, Kolding*, 2000.

[11] D. Salber, A. K. Dey, and G. D. Abowd. The Context Toolkit: Aiding the development of context-enabled applications. In *Proc. of the SIGCHI Conference on Human Factors in Computing Systems: The CHI is the Limit*, 1999.

[12] H. Schmid, A. Fitschen, and U. Heid. Smor: A german computational morphology covering derivation, composition, and inflection. In *Proc. of the 4th International Conference on Language Resources and Evaluation (LREC 2004)*, Lisbon, Portugal, 2004.