

# Geospatial Information Processing in WG A1

Bernhard Lorenz, Hans Jürgen Ohlbach, Edgar-Philipp Stoffel

## I. INTRODUCTION

WG A1 has two main working hypotheses with respect to geospatial information processing for the Semantic Web.

- 1) In order to give information on the Web its real semantics, i.e. to realize the *semantic* web, we must develop models of a considerable part of the reality. In our case this amounts to modelling geographic and other spatial information in such a way that it can be used in a flexible way by many different applications.
- 2) The Semantic Web will not be confined to desktop computers in offices and homes. There is an enormous potential of new applications when Ubiquitous Computing and the Semantic Web eventually merge. The Semantic Web will be accessible from Personal Digital Assistants (PDAs), mobile phones, wearables and many other computing devices. Since most of them are mobile, it is of particular importance to provide them with a geospatial model of the reality. Therefore certain developments in WG A1 target applications which run on mobile devices.

The main bulk of geospatial information is static. Road and train networks, for example, do not change that often. There is, however, a small, but important part of dynamic geospatial information. News about traffic jams, train delays, etc. can be very important and therefore must not be neglected. A comprehensive geospatial model must therefore combine static and dynamic information.

Another important aspect is the granularity of the geospatial information. In order to navigate a car through a city, it is necessary to have a very detailed road map. If, on the other hand, one wants to list all major cities in Europe, it is sufficient to know that, in particular, Munich is in Germany, Germany is in Europe and 'is in' is transitive. Since Semantic Web applications may be of almost any type, the whole range from very detailed coordinate computations up to very abstract spatial reasoning should be available.

Many geographic notions are very much application dependent. "City A is between city B and city C", for example, may be true if you consider a train network, where the train line makes a detour through city A. It may be false if you consider a highway network where the closest highway is very far from city A. Notions like "between" and many others can therefore not be hard coded into a geospatial system. Instead, it must be possible to specify them in a suitable specification language such that one can use them for computation and reasoning tasks.

This research has been co-funded by the European Commission and by the Swiss Federal Office for Education and Science within the 6th Framework Programme project REVERSE number 506779 (cf. <http://reverse.net>)

The systems we are developing in WG A1 try to address all these aspects:

- 1) representation and computation with static and dynamic geospatial information;
- 2) combination of detailed geographic computation with abstract spatial reasoning;
- 3) the possibility to specify application dependent geospatial notions in a suitable specification language.
- 4) Finally, it is necessary for testing the components, and eventually for many applications, to visualise geospatial information or to output it in another way (written or spoken language, for example). Therefore rendering components of different kinds are also an important aspect in our work.

## II. LOCAL DATA STREAM MANAGEMENT SYSTEM

The traditional way in which data is managed and made available for processing is via (relational) *Database Management Systems* (DBMS), which expect data to be put into the form of *persistent data sets*. Whereas for many applications this constitutes a suitable form of data storage, there exists a growing number of applications which require data to be treated and processed as a *continuous stream* [1]. Currently a number of different *Data Stream Management Systems* (DSMS) are developed to facilitate access to data streams.

The *Local Data Stream Management System* (L-DSMS) which was developed by WG A1 is *local* in the sense that it facilitates the specification and construction of a single Java program which consists of a network of nodes for processing streams of data. Each such node receives data from one or several data sources, processes them in a certain way, and delivers the processed data to one or more data drains. A data drain can be the data source for the next processing node in the network, or it can be the end application in the whole processing chain. One of the components of L-DSMS is the SPEX XML-filtering system [2], [3]. It processes XPath [4] queries on a stream of XML data and can be used to extract interesting information from XML streams.

L-DSMS has been developed and tested in an application for processing dynamic traffic information. The traffic information comes from RDS-TMC receivers, is converted to an XML stream, is subsequently processed in several steps by the L-DSMS and then delivered to other application systems, one of which is described in section III. Further details about L-DSMS can be found in [5].

Fig. 1 shows (from left to right) different examples of configurations of node networks, which (1) receive the stream from a socket connection for output on a screen and logging on disk, (2) are directly connected to an RDS/TMC receiver and

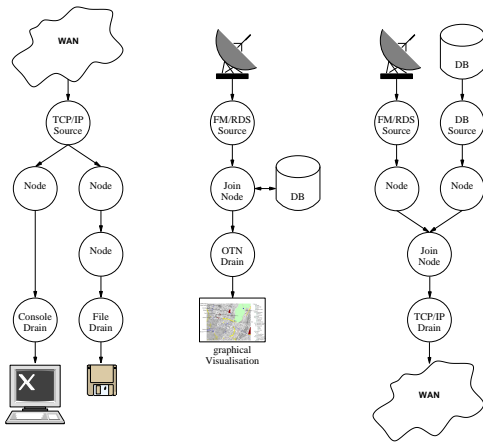


Fig. 1. Examples of L-DSMS configurations

enrich the stream by data from a relational database before providing output on a graphical map in a browser window, and (3) join a stream from a receiver with an artificial stream generated from data in a relational database and providing the joint stream for use by another application via a socket connection.

### III. KML TRAFFIC INFORMATION SERVICE

This project serves to join several components into an application providing real time traffic information for display in an online map system, in this case *Google Earth*. The name of the service is derived from *Traffic Message Channel (TMC)*, an ISO standard for traffic information and the source for the transformation, and the *Keyhole Markup Language (KML)*, a language developed by Keyhole Inc., now part of Google, which is used to provide data for the Google Earth client. As sketched above, Traffic and Travel Information (TTI) is collected via suitable RDS/TMC receivers and transformed into an enriched XML stream. This stream can optionally be transformed and filtered by L-DSMS before it is read by the TMC-KML transformation system.

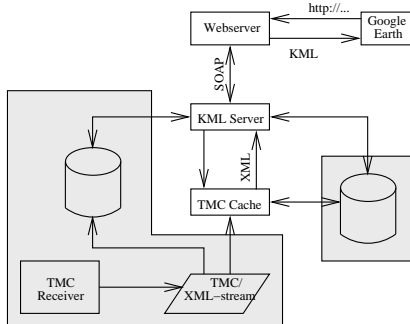


Fig. 2. TIS System Architecture

As shown in fig. 2 the Traffic Information Service (TIS) consists of three main components: a *TMC Cache Drain*, the *KML server*, including a relational database for static information, and a *web server*. The shaded grey components are parts of L-DSMS which were developed prior to the

development of TIS. The databases store static information, such as the Location- and Event Code Lists needed for TMC.

The *Cache Drain*, closely connected to L-DSMS, serves to keep track of current TTI messages. This is necessary because the TMC system is based on the recurring broadcast of traffic messages, including massive repetitions in order to reduce latency and improve the timeliness of the messages. The cache drain retains a list of all currently active messages, adds new messages as they come in via the XML stream, and gets rid of outdated messages which either reach the end of their life cycle or are explicitly invalidated by special cancellation messages.

The *KML server* waits for external requests coming in via the web server and generates KML documents from the cached TMC data on-the-fly. These documents are then made available via the *web server*.

In a closely related project we collect TTI over longer periods of time for the purpose of developing statistical data about traffic networks. This is rather important for routing applications such as described in section V because the throughput in traffic networks is highly depending on the individual load on each of its connections, i.e. the different states of congestion. Time dependent data about congestion enables the application for example to propose a different routes between identical locations *A* and *B* at certain times, for example during and outside of rush-hours, on holidays, etc. This is possible because the applications can statistically “predict” that on a certain connection there is a high chance for congestion between e.g. 7am and 9am and again between 4pm and 6pm.

### IV. MULTI PARADIGM LOCATION LANGUAGE

Geographical distances among geospatial objects are expressed by proximity relations, such as “A is near B”, “C is far from D”. There exist a number of approaches to model proximity relations in the geospatial domain. Within the semantic web context we propose an approach based on multi modal path planning, since computing semantically correct distances almost always amount to path planning problems. This is due to a number of reasons, chiefly because  $L_1$  distances do not reflect people’s intuitive understanding of distances. See section V for more detail on our approach.

A fuzzy logic approach for proximity reasoning for instance is proposed by [6]. Proximity expressions such as *far* or *close to* have a corresponding fuzzy membership function. Using this model, the query “Which shopping centres are close to *R*” takes the following form:

$$Close : O = CloseTo\{o : O, \{o\}, R, \{x_1, y_1, x_2, y_2\}, DistanceMethod, C\} \quad (1)$$

$O$  is the object type (shopping centre), *CloseTo* is a fuzzy set membership function, and *DistanceMethod* is a distance calibration method (e.g. absolute or relative distance). Object  $o$  is an object of type  $O$ ,  $R$  is the reference location, and  $\{x_1, y_1, x_2, y_2\}$  serves to represent scale by denotes the area. Since proximity depends on contextual information, the context  $C$  also has to be given. It includes factors such as

mode of transportation, user profile and preferences, possibly device characteristics and more. See below for more detail on context and user modelling. Since most transportation modes are based on network like structures, they have a significant impact on the notion of proximity. Therefore, two objects might be regarded as *near* to each other in one context, but *far* in another. In our example, the result of the query above is a set of objects that is *close* to  $R$  and is of type  $O$ .

In MPLL spatial relations cannot be entirely hardcoded. On the contrary, spatial relations have to be user definable, so that from a set of predefined as well as user-defined relations new relations can be composed. MPLL will therefore provide basic quantitative relations, such as those based on angular and distal values, as well as qualitative relations and interpretations thereof. Based on angular values, the relation “*between*” for example could be defined as follows,  $\theta_{AB}$  being the bearing between points  $A$  and  $B$ . If

$$|\theta_{AB} - \theta_{AC}| = 180^\circ \pm 5^\circ \quad (2)$$

then the statement “*A is located between B and C*” is true<sup>1</sup>. Optionally, this relation can be extended to support fuzzy logic by mapping the (limited) deviation from the core value  $180^\circ$  (i.e. the interval of the support values around the core value) to values in the interval  $[0, 1]$ .

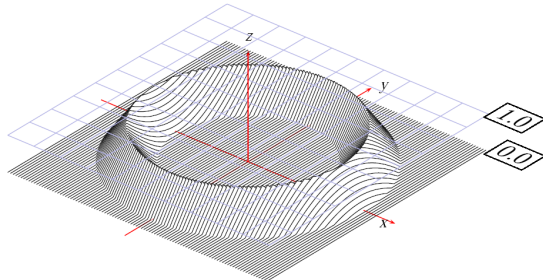


Fig. 3. 2-dimensional Fuzzy Distance

The same approach can be used for distal relations. An omnidirectional fuzzy distribution around a point in space is shown in fig. 3. For each point in planar space, i.e. each pair of coordinates, the value on the  $z$  axis represents the fulfilment of the distal relation.

## V. TRANSROUTE: A FRAMEWORK FOR GRAPHS

Focal points of this component are both representation and manipulation of graphs in a programming environment. A large variety of algorithms ranging from connectivity and clustering over flows to shortest path computation are provided by *TransRoute* [7].

In particular, graphs are considered as first-class generic data types as part of a graph library, each of which can be instantiated with concepts of a domain ontology, and also

<sup>1</sup>This refers to a navigational context, i.e. regarding free movement in space. In a network based environment, as mentioned in the introduction, this approach might not be suitable.

decorated by according attributes. For this purpose, the open-source *Java Universal Network and Graph* framework (*JUNG*) [8] based upon the object-oriented paradigm and its relevant design patterns has been extended with respect to following functionalities:

### A. Persistence of Data

For any application dealing with high volumes of data, storing these solely in transient memory is not only unsatisfactory but inconceivable. Instead, making them persistent is the passable solution. In order to overcome this problem in *TransRoute*, the *Graph eXchange Language (GXL)* [9] defined in an XML vocabulary is employed for exchanging graphs. Consequently, both storing and retrieving graphs from a persistent repository are possible. The key advantages using *GXL* are revolving its advanced data model, in which e.g. nested graphs and attributes of various basic datatypes are included.

### B. Hierarchic Graphs

Special emphasis is put on cluster-like hierarchic structures [10], [11], [12], [13], [14] which essentially compose geospatial networks.

The basic idea behind is that graphs of a particular level are being condensed to one node of the level above. The former can also be seen in terms of clustering, i.e. compressing a graph to one node is a natural grouping into a logical (or sometimes physical) cluster.

By adopting a high-level viewpoint of transport networks (represented as a vertex on this abstract level), they can not only be matched to regions, but also to network types indicating the modes of transportation which may be used. The essential idea focusses on employing one network for each mode of transportation as a separate graph, which fits well into our hierarchic concept.

### C. Ontologies

Ontologies have proven to be well-suited to describe complex entities and features of a modelled domain in a machine-processable way. For our purpose, *TransRoute* uses the *Ontology for Transport Networks (OTN)* [15] which has been developed in the *Web Ontology Language (OWL)* [16]. *OTN* is used for the abstract modelling layer for representing domain concepts. Graphs, vertices and edges can be created by specifying the domain concept they ultimately stand for, thus opening the bridge to domain-specific knowledge.

### D. Interfaces for Algorithms

Interfaces are an elegant way for conveying the idea of design by contract: Since there are situations in which one can choose from different algorithms solving the same problem, the strategy pattern has been applied: all algorithms (plugins) are realising the same contract imposed by the interface *IAlgorithm*. Hence, properties of different algorithms can be compared.

### E. Application

In a first **case study** based on the metropolitan street network data of Munich, encoded in an *OWL* instance file containing a total of 13300 vertices and 19887 edges, the algorithms of *TransRoute* could be tested on real data. These preliminary results are promising, yet further data of a larger magnitude (for Germany) will be taken into account for testing efficiency in practical application.

### REFERENCES

- [1] S. Babu and J. Widom, "Continuous Queries over Data Streams," *SIGMOD Record*, vol. 30, no. 3, pp. 109–120, 2001.
- [2] D. Olteanu, "Evaluation of XPath Queries against XML Streams," Dissertation/Ph.D. thesis, Institute of Computer Science, University of Munich, 2005, PhD Thesis, Institute for Informatics, University of Munich, 2005. [Online]. Available: <http://www.pms.ifi.lmu.de/publikationen/#PMS-DISS-2005-1>
- [3] D. Olteanu, T. Furche, and F. Bry, "An efficient single-pass query evaluator for XML data streams," in *SAC*, 2004, pp. 627–631.
- [4] J. Clark and S. DeRose, "XML Path Language (XPath) Version 1.0," W3C – World Wide Web Consortium, Recommendation, 1999, <http://www.w3.org/TR/xpath>.
- [5] H. J. Ohlbach and B. Lorenz, "Dynamic Data for Geospatial Reasoning - A Local Data Stream Management System (L-DSMS) and a Case Study with RDS-TMC," 2006.
- [6] M. Gahegan, "Proximity operators for qualitative spatial reasoning," in *COSIT*, 1995, pp. 31–44.
- [7] E.-P. Stoffel, "A research framework for graph theory in routing applications," Diplomarbeit/diploma thesis, Institute of Computer Science, LMU, Munich, 2005. [Online]. Available: [http://www.pms.ifi.lmu.de/publikationen/#DA\\_Edgar.Stoffel](http://www.pms.ifi.lmu.de/publikationen/#DA_Edgar.Stoffel)
- [8] "Sourceforge.net jung: Java universal network/graph framework." [Online]. Available: <http://jung.sourceforge.net>
- [9] "Gxl: Graph exchange language." [Online]. Available: <http://www.gupro.de/GXL/>
- [10] A. Car, H. Mehner, and G. Taylor, "Experimenting with hierarchical wayfinding," 1999. [Online]. Available: [citeseer.ist.psu.edu/car99experimenting.html](http://citeseer.ist.psu.edu/car99experimenting.html)
- [11] P. Eades and Q.-W. Feng, "Multilevel visualization of clustered graphs," in *Proc. Graph Drawing, GD*, no. 1190. Berlin, Germany: Springer-Verlag, 18–20 1996, pp. 101–112. [Online]. Available: [citeseer.ist.psu.edu/eades97multilevel.html](http://citeseer.ist.psu.edu/eades97multilevel.html)
- [12] D.-I. M. Rose, "Modeling of freeway traffic," in *10th International Conference on Computing in Civil Engineering*, Weimar, June 02-04 2004. [Online]. Available: <http://www.bauinf.uni-hannover.de/publikationen/ICCCEBPaperRose.pdf>
- [13] B. Riedhofer, "Hierarchische strassengraphen," 1997, master Thesis. University of Stuttgart, Faculty of Computer Science. [Online]. Available: [elib.uni-stuttgart.de/opus/volltexte/1999/7/](http://elib.uni-stuttgart.de/opus/volltexte/1999/7/)
- [14] G. Busatto, "An abstract model of hierarchical graphs and hierarchical graph transformation," Ph.D. dissertation, University of Paderborn, 2002.
- [15] F. Ipfelkofer, "Basisontologie und anwendungs-framework für visualisierung und geospatial reasoning," Diplomarbeit/diploma thesis, Institute of Computer Science, LMU, Munich, 2004. [Online]. Available: [http://www.pms.ifi.lmu.de/publikationen/#DA\\_Frank.Ipfelkofer](http://www.pms.ifi.lmu.de/publikationen/#DA_Frank.Ipfelkofer)
- [16] "W3c owl: Web ontology language." [Online]. Available: <http://www.w3.org/TR/owl-features/>