

Supporting Multi-Modal Interfaces for Adaptive Tour Planning

Häußler Jochen, Zipf Alexander

European Media Laboratory GmbH(EML), Schloss-Wolfsbrunnenweg 33 69118 Heidelberg
E-mail:jochen.haeussler@eml-d.villa-bosch.de

Abstract

Typical examples for mobile GI services include tour planning and maps for navigation support (Reuter, Zipf, 2004). We have developed several prototypes of mobile GI services for navigation tasks including innovative services like user-adaptive tour planning. We are also among the first to support multi-modal (i.e. graphical, natural language and gestures based) user interfaces for such GI related mobile services in several projects like Deep Map, Crumpet and SmartKom since several years. As the development of GI services that adapt to user and context parameters as well as multi-modal interfaces for mobile navigation support are relative new research areas for GIScience it was necessary to specify interfaces for the software components that have been developed. These-relevant parts of the so-called Deep Map Objects (DMO), the XML-based message objects of the Deep Map multi agent system-will be introduced in this paper. But only very recently also the OGC Open Location Services Initiative (OpenLS) has published a specification for similar services in the area of LBS-but yet without a focus on adaptation or multi-modality. Both representations include relevant data types for route planning and map generation and are based on XML. Therefore we will compare these two representations and discuss their applicability for developing multi-modal map interaction for adaptive tour planning.

Keywords

Multi-modal user interfaces, OGC OpenLS, Tour Planning

I. INTRODUCTION

The adaptivity of GI services to context can be seen as one of the next steps for GIScience research in order to achieve more intuitively usable GI services. Especially the use of GI services in mobile environments goes along with a steady change of context in which the interaction between user and system occurs. We assume that apart from ubiquitous availability and context adaptivity the kind of human computer interaction is a third subject that needs to be considered when developing future ubiquitous GI services. This view is for example supported by a field evaluation of the SmartKom Mobile prototype which showed that in mobile environments a multi-modal interaction including speech and pointing gestures on small handheld devices is well accepted (Malaka, et al., 2004).

The recently published Open Location Services by the Open Geospatial Consortium (OGC) is an important step for our vision of ubiquitously available GI services. In consequence we investigate the applicability of the OpenLS Core Services for ubiquitous and context aware GI systems with support for the requirements of multi-modal systems.

In this paper we can only present hints as which adaptive services might be suitable within the context of GI services. This is derived from first results regarding adaptive mobile GI services in some of our projects, mainly Deep Map (Zipf, 1998), CRUMPET (Poslad, et al., 2002) and SmartKom (Wahlster, 2004). Leaving out of consideration the parameters a system should adapt to we want to focus on categories that are affected by context adaptivity. The following categories of adaptation of mobile GI services can be identified:

- adaptation of route planning (by individual weighting and restrictions)
- adaptation of the offered contents (e.g. concerning detailedness, topic) – including the location-dependent selection of content
- adaptation of the visual presentation of the contents offered (pictures, maps, video, VR)

Examples on adapting maps are presented in (Zipf, 2002; Meng, et al., 2004), the context- and user-dependent adaptation of spatial queries is presented in Zipf and Aras (2002). Within this paper we will focus on the personalized route planning example in the beginning and then discuss how to support new interaction possibilities like multi-modal input for LBS.

Some of the most important basic data types for a location-aware tour guide are *Location* and *Tour*, as well as *Geometry* and *Features*. As the Open Geospatial Consortium has standardized an XML-representation for geographic features and geometry called Geographic Markup Language (OGC, 2001). GML is the base for our geometry types as well. But until recently GML lacks richer semantics like the definition of routes and tours. Within this paper we focus on the XML-based definition of *Tours* and *Routes*. These are of particular importance when guiding users through an area, as they ideally include the knowledge about important features of the tour. These are for example of particular importance to give useful directions in natural language or to produce sophisticated tour maps.

II. ROUTE PLANNING

In usual route planning applications a route server computes a path through a street network, given two or more locations on the network - typically the fastest or shortest. In cases where street addresses are specified as either the starting or ending points of the route, an address geo-coding service is required, that resolves the addresses.

In our projects on mobile navigation support for tourists (Deep Map, CRUMPET, SmartKom) we have developed a *tour agent* which does exactly that job, but it has been realized as an agent in a multi agent system instead of a web service. It is termed tour agent because in Deep Map we distinguish a simple *route* from one point to another by one transportation means of transportation from a more complex *tour* which may be composed by multiple routes. Tour requests (in form of xml-messages) can be sent to the agent including hard constraints like starting point, destination and potentially waypoints. The task of the tour agent is just to calculate a path through the street network. This simple tour agent is comparable with the OpenLS Route Service which is an interface specification for the very same purpose. We first discuss the interfaces of the Deep Map tour agent – known as DMO (Deep Map Objects) schema – in section II.A and then in section II.B we give a short introduction to the OpenLS route service and compare the interfaces of both afterwards.

In addition to that standard tour agent we have developed a *tour proposal agent*. While the tour agent provides the well known routing functionalities just mentioned, the task of the tour proposal module is much more complex. Its task is to suggest individual sight seeing tours to visitors of a region or

a city according to the user’s current interest and further context information (Zipf, 1998). This will be introduced in section II.D.

A. The interfaces of the deep map tour agent

The interfaces of the Deep Map tour agent are simple: There are only one request type and one response type within the DMO schema:

The request type *GTAComputeTour* is shown in Figure 1. This figure is a visual representation of the relevant part of an XML schema. The tree shows how the elements are nested: on the left is the top level element for this request, *GTAComputeTour*. It consists of a sequence of elements: A start time (optional), a start location and a destination location (*endLocation*), optional waypoints (*locationToVisit*). The data type of *locationToVisit* is composed of the general *location* type of the DMO schema (which is used on many places in the entire schema) plus a visit duration. This time duration has to be used by the tour planning implementation to calculate the duration of the complete tour. With the element *meansOfTransportation* optionally a transportation modality can be specified. Allowed values are bike, wheelchair, car, cycle and foot. The Deep Map tour agent defaults the transportation mode to pedestrian navigation (value: foot). This means of transportation will be used for the entire tour. So if there should be used several different means of transportations on the same trip it would be necessary to split this trip in parts with a single means of transportation.

The Boolean field *preserveOrder* specifies whether the order of the waypoints which are specified as *locationToVisit* has to be kept, or this order may be optimized in the calculated route. The element *tourIntention* can be used to choose

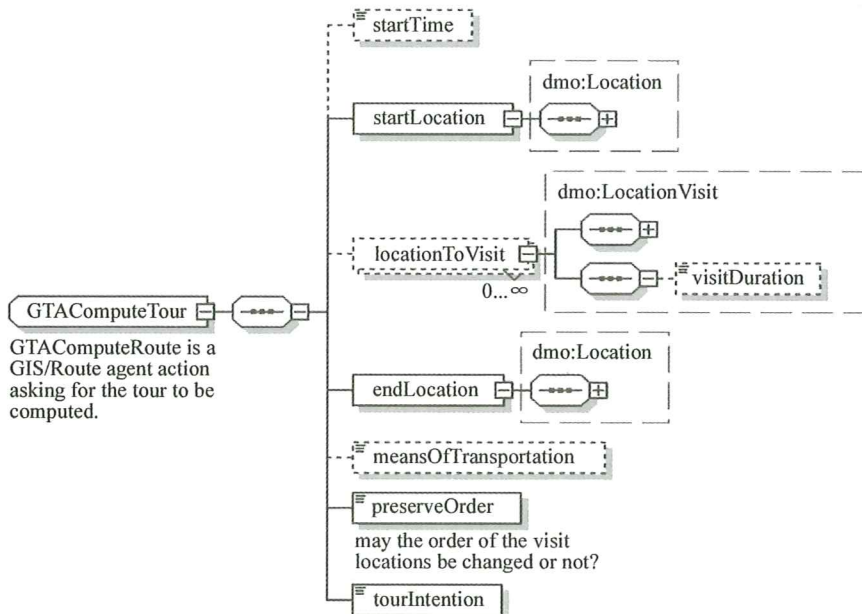


Figure 1. Request schema for the tour agent asking for calculation of a tour

between the shortest path or a sightseeing tour. In fact, this value in a route request is relevant for the decision to which agent – tour agent or tour proposal agent – the tour planning task is delegated.

The response type for the tour request is *GRTour* (Figure 2). *GRTour* is a complete description of a tour, including the whole geometry of the path, complete location objects as start point and visit locations and additional information like duration, length etc. Each calculated tour gets its own identifier *tid*. This can be used to request previously calculated tours from the agent, which stores a history of all tours.

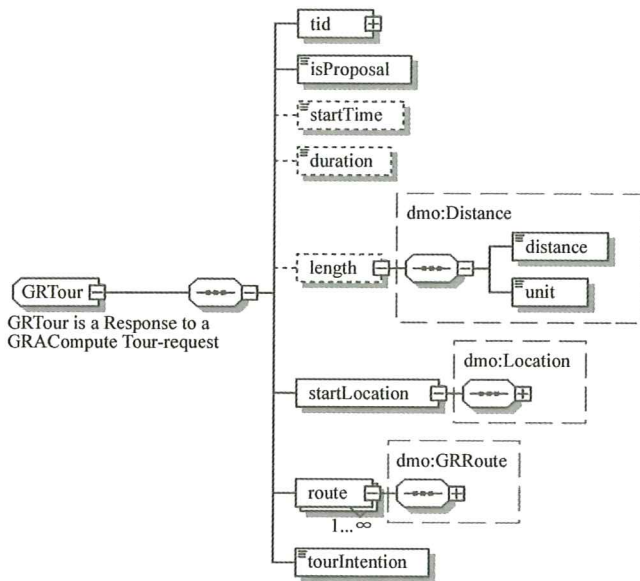


Figure 2. Structure of the response schema for the tour agent describing a tour

A *GRTour* object contains one to many elements *route*. The data type of this element is the complex type *GRRRoute* (Figure 3). This represents the path from one waypoint to the next one. The *visitLocation* of a *GRRRoute* represents the destination of this path. The starting point of a *GRRRoute* is the *visitLocation* of the previous *GRRRoute*. In the case of the first route element of a *GRTour* it is the *startLocation* of the entire *GRTour*. The *endLocation* that was specified in the request will be contained in the response as *visitLocation* of the last *GRRRoute* element. This was modelled that way to avoid locations (which are nested structures themselves) to occur redundantly in a *GRTour* message.

But the representation of the route is nested even deeper (compare Figure 4): Every *GRRRoute* element is itself composed of *GRRRouteElements* (Figure 5). This is the smallest unit of a

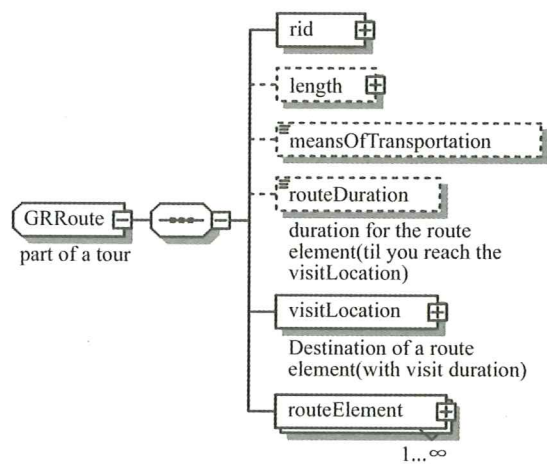
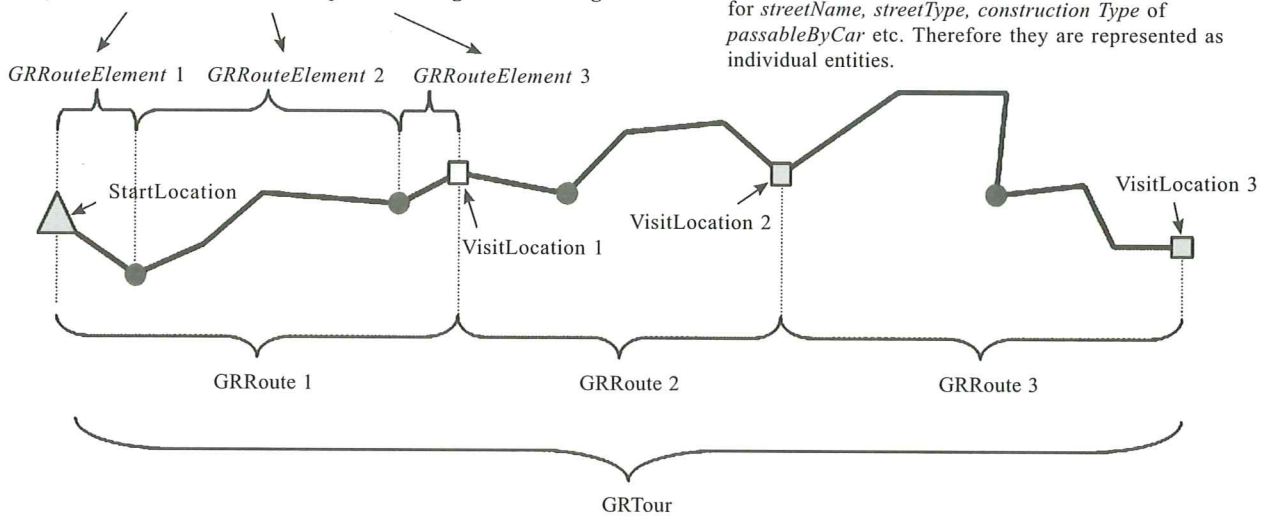


Figure 3. Each *GRRRoute* represents the route from one waypoint (*visitLocation*) to another. A *GRTour* is composed of 1...n elements of the type *GRRRoute*

Geometry of each *GRRRouteElement* represented as *gml:LineString*



GRRRouteElement 1,2,3 have different properties-e.g. for *streetName*, *streetType*, *construction Type* of *passableByCar* etc. Therefore they are represented as individual entities.

Figure 4. Schematic representation of a *GRTour* consisting of *GRRoutes* made from *GRRRouteElements* represented as *gml.LineStrings* for sections with the same attributes

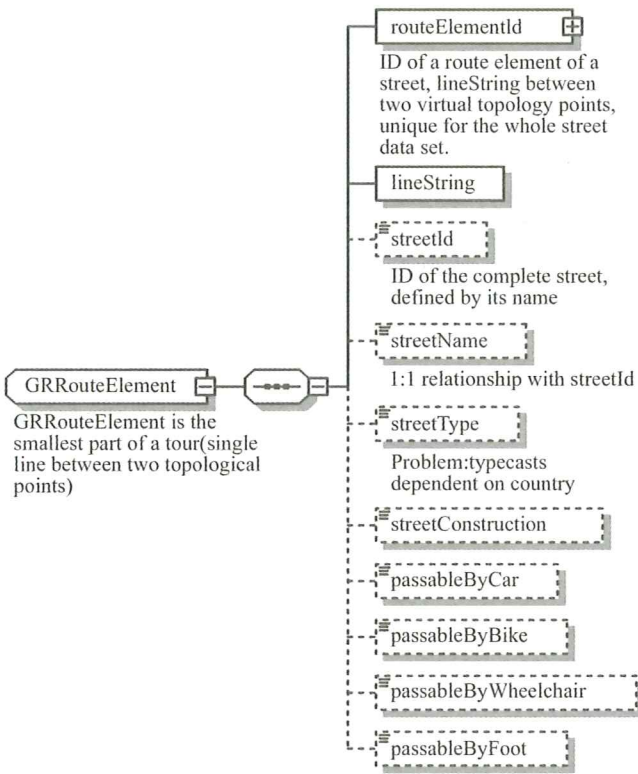


Figure 5. *GRRouteElement* is the smallest part of a tour resp. route. It represents an edge in the graph of the underlying street network

calculated it and whether it is a simple route from one point to another or a complex tour with several waypoints and potentially different means of transportation. This can be expressed with the *GRTour* element, but it is not implemented in the current Deep Map system. The resulting *GRTour* object contains all information that is necessary to describe the tour, to show it on a map or to do other calculations on it like i.e. segmentation for an incremental guidance or updating the dynamic lexicon of a speech recognizer as done in the SmartKom system.

B. A short overview of the OpenLS specification

As we have presented the basics of the tour interfaces developed in our projects we now want to introduce the OpenLS specification. The architecture is called GeoMobility Server (Figure 7) and comprises core services like route services as well as presentation services (in particular maps). The OpenLS Information Model (Figure 6) is comprised from Abstract Data Types (ADTs) which we present shortly (OpenLS, 2003):

Address ADT contains address information for a geographic place. The Address ADT consists of a street address (or intersection), place name, postal code, street locator, building locator, and supplemental address information.

Area of Interest (AOI) ADT contains an Area of Interest as defined by a named circle, bounding box, or polygon.

Location ADT is the extensible, abstract type for all expressions of Location that can be used by OpenLS application and services to specify the location of a target or a subscriber. The element Location is the root of a tree that includes Point, Position ADT, Address ADT, and POI ADT as its subtypes.

Map ADT contains a rendered map that results from the Map Portrayal Operation of the Presentation Service. See paragraph 3.

Point of Interest (POI) ADT includes a place or entity with a fixed position that may be used as a reference point or a target

Deep Map tour representation, describing the path from one node in the underlying topology to the other. The *GRRouteElement* contains some important attributes of this edge like street name, other street parameters and the geometry. The geometry is specified by a GML line string.

In the end we can say that in Deep Map there is only one request type for tours. Depending on the content of such a request, the appropriate agent becomes active and computes the result. The result-a *GRTour* object-is a complete geometric and semantic description of the calculated tour. The structure of a tour is always the same, no matter which agent has

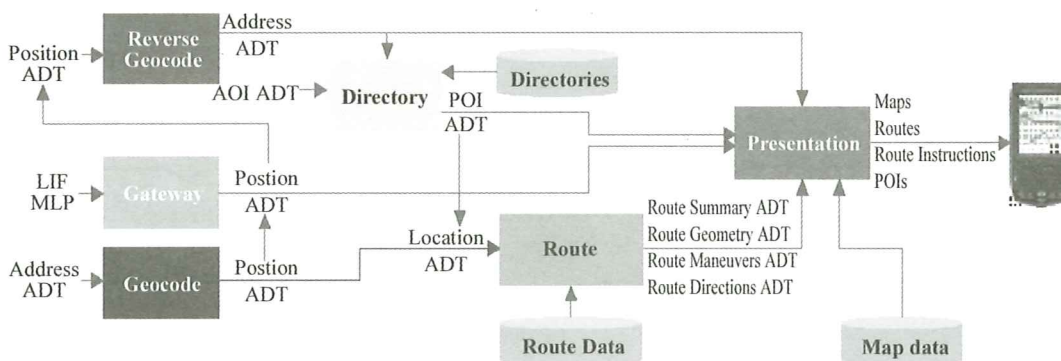


Figure 6. The OpenLS information model (OpenLS 2003)

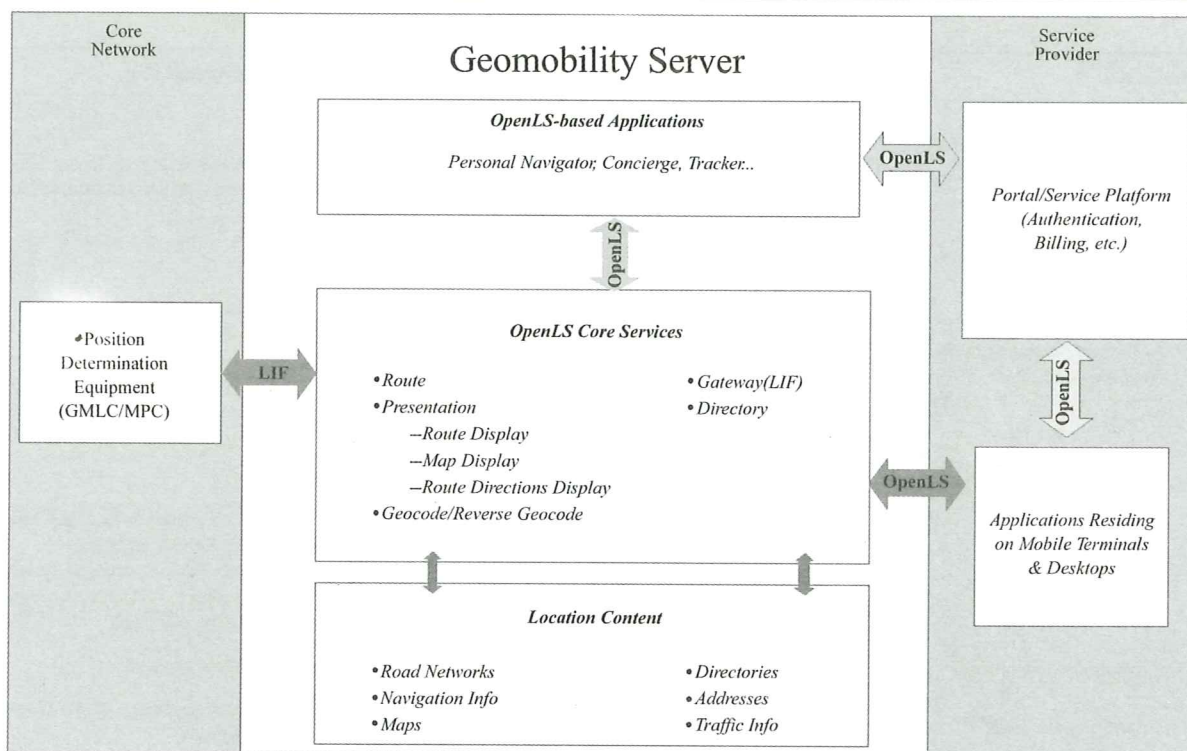


Figure 7. The OpenLS GeoMobility server(OpenLS 2003)

in an OpenLS service. It contains name, type, category, address, phone number, and other directory information about the place, product, and/or service.

Position ADT contains any observed or calculated geographic position and quality of position and relates to the *Location*, *Shape* and *Quality of Position* elements of the Mobile Location Protocol (MLP) Specification (Version 3.0, OMA).

Route Instructions List ADT contains a list of travel instructions consisting of turn-by-turn directions and advisories along the route, ordered in sequence of their occurrence, and formatted for presentation to the user.

Route ADT consists of two ADTs: Route Summary and Route Geometry. Route Summary contains the route's overall characteristics (start point, waypoints, end point, transportation type, total distance, travel time, and bounding box). Route Geometry contains a list of geographic ordered positions along the route, as well as the geometry of the route segments as list of points.

C. A comparison between openLS and the DMO specification for routing applications

In the following we want to compare the route requests of the OpenLS specification and the Deep Map interfaces. Of course the requests for each system mirror its functionality. Therefore it is clear that there are some basic differences. First of all the respective systems have different architectures: The OpenLS

Core Services are web services and the Deep Map system is a multi agent system (www.fipa.org). But both share a set of comparable functionalities and we found that the responsible interfaces are astounding similar.

Table 1 presents a complete comparison of functionalities of both specifications. The respective root elements for route requests are called *DetermineRouteRequest* (OpenLS) and *GTAComputeTour* (DMO). We now want to discuss aspects we consider important in more detail: One of them is how good the (route) requests are suited to determine the result of the calculation. Two points seem important:

- (1) How good can the course of the route be determined by the client?
- (2) Is there a possibility to select the appropriate response type and in what granularity?

Both systems are very similar regarding the first issue. They both allow to specify start time and start location together with a list of waypoints. Only DM allows specifying whether the order of waypoints has to be respected or may be optimized. Only OpenLS allows specifying types of features, locations and areas in which the route should avoid passing through. The possibility to select a means of transportation and to select a fastest or shortest route differs a little, but is available in both specifications. The possibility to use real time traffic information is still missing in the Deep Map specification.

Table 1. Comparison of *GTA compute tour* and *determine route request*

Functionalities	OLS:DetermineRouteRequest	DMO:GTAComputeTour
Start time	<i>expectedStartTime</i> (Attribute of RoutePlan)	startTime
Possibility to store route server side	<i>provideRouteHandle</i> (Attribute of DetermineRouteRequest)	Routes are stored by the map agent automatically and can be requested with <i>GTAGetTour</i>
Specification of distance units	<i>distanceUnit</i> (Attribute of DetermineRouteRequest)	(not possible)
Reference to Route that has been calculated before, but being actualized	RouteHandle - Reference to a previously determined route. The new route may be different than the existing route due to changes in real time traffic, and if the travel start time defaults to the current time, due to transportation network time restrictions, or any other reason.	With <i>GTAGetTour</i> a Route stored on the server can be requested. However, this route will not be recalculated with e.g. actual traffic information. To get a recalculated tour a new <i>GTAComputeTour</i> request has to be performed.
Possibility to use real time traffic information	<i>useRealTimeTraffic</i> (Attribute of RoutePlan)	(not possible)
turn-by-turn description of the route	RouteInstructionsRequest with mime type (text, voice etc.)	Functionality is available in the Deep Map system as specific request (SAGuide with <i>incremental=false</i>) to a specialized agent for incremental guidance, but cannot be requested in an interface of the tour agent.
Geometry of the complete route	RouteGeometryRequest	Is automatically responded.
Map(s) visualizing the route	RouteMapRequest	Separate request necessary - <i>GMAComputeMap</i> for theMapAgent
Possibility to request only some information types	Elements RouteInstructionRequest , RouteGeometryRequest and RouteMapRequest are optional. Type of required information can be requested exactly.	Response type to a <i>GTAComputeTour</i> request is a complete <i>GRTour</i> which cannot be adapted to the request.
Specify fastest/shortest etc.	RoutePreference	tourIntention - we can distinguish shortest path and sightseeing tour. The shortest path is calculated by the simple tour agent. A sightseeing tour can be adapted to user preferences and is calculated by the tour proposal agent.
Specify means of transportation	RoutePreference (enumeration values: Fastest, Shortest and Pedestrian. It is not specified which means of transportation should be used when Fastest or Shortest is specified in the request)	meansOfTransportation (enumeration values: undefined, wheelchair, car, foot, bike, motorBike, car, taxi, public. Undefined could be matched to an application-specific default, in deepmap: foot. Public transport has not been implemented in Deep Map)
Specification of waypoints along the route	WayPointList	locationToVisit (location + duration for a visit. The computed route will be segmented from one location to another)
Specify order of waypoints	(not possible)	preserveOrder (boolean)
Specify types of features, locations, areas in which the route should avoid passing through	AvoidList (enumeration of features to avoid: Highway and Tollway)	(not possible)

More differences can be found in the second point: As explained in paragraph 2.1 the DMO reply to a route request is a complete *GRTour* object. This is a very complex and sometimes quite lengthy structure. In some use cases this may be overhead - e.g. when a client just wants to see the route visualized on the map, without the need for more advanced features - like multi-modal interaction. Here we see advantages of the OpenLS Specification. There it is possible to specify in detail which information is needed by the client by using the elements *RouteInstructionsRequest*, *RouteGeometryRequest* and *RouteMapRequest*. The Route

Service can this way return any combination of:

- summary information
- route geometry
- maps of the route
- turn-by-turn instructions and advisories for presentation

The resulting route can also be optionally stored on the terminal or application server. The user may store it for as long as needed, thus requiring the means to also fetch a stored route.

It seems quite interesting that both – “maps of the route” and

“turn-by-turn instructions of the route” – are within the responsibility of the OpenLS Route Service, while in the DM approach separate entities are responsible for this tasks. We believe that this allows a more fine grained adaptation and flexible usage because it makes it possible to put the parts together in a manner most suitable for the individual scenario. More important seems the possibility to put more advanced adaptation engines between the individual services in order to deliver aggregated applications to the end user.

Another issue is the support for developing applications that offer incremental guidance which includes for example the segmentation of paths presented by the route service into information chunks more suitable for the current situation as subtasks. Path segmentation is a key process related to trajectories, routes, and directions. In Deep Map the incremental guidance is realized as collaboration work of several agents with several interfaces (Kray, 2003). Therefore it can hardly be compared to the OpenLS turn-by-turn response.

D. Personalized tour proposals – an advanced routing task

A system that is capable of generating individual tour proposals through a city based on the personal preferences and interests of a tourist needs to solve several problems including the application of individual interests of the user. If these are known, there are several possibilities how to include them into a tour planning or tour proposing algorithm. First of

all the range of possible attributes that may influence the choice for a particular section of a route have to be identified and modelled. Appropriate variables have to be included into the database and attached to the street network. Such attributes include both “hard” restrictions, or physically given attributes (like height, steepness, turn rules, legal rules, etc.) as well as a range of more dynamic and “soft” parameters. Their importance may vary considerably from person to person or within time. Such parameters could include aesthetic aspects, the social milieu of the area, dislike of motorized traffic or preferences for areas with particular architectural elements or a high rate of nice viewpoints. Another task of such a tour proposal is to find a reasonable number of waypoints that seem to match the user’s interests very well and are reachable within the users time constraints. The maps in Figure 8 present the results of an algorithm implemented in a first prototype using standard GIS software when varying the degree of importance for particular soft parameters.

Based on this work further algorithms were developed and incorporated in the tour proposal agent (Jöst, Stille, 2002). It finds user-optimal tours respecting hard time constraints. The underlying mathematical model is the so-called “profitable tour problem” which is closely related to the prize-collecting travelling salesman problem (Ballas, 1989), a generalization of the travelling salesman problem (TSP). Since these problems are NP-hard, exact solutions require too much computational effort for exact solutions, so heuristics have to be used.

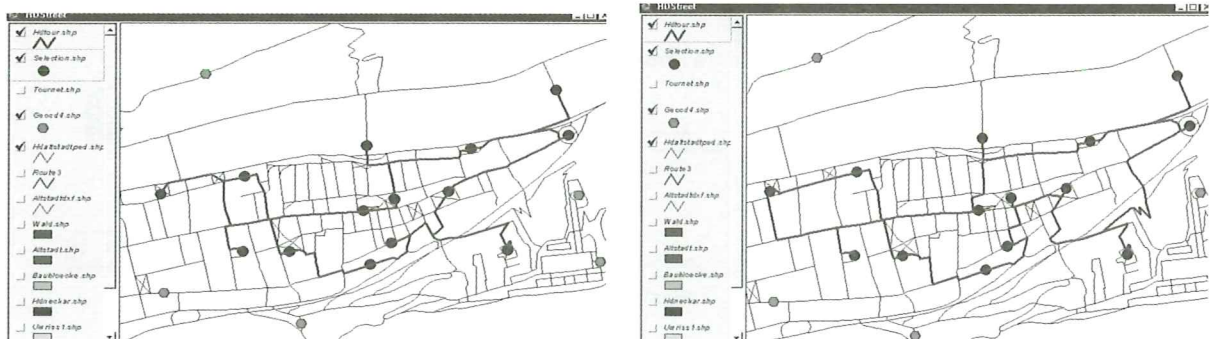


Figure 8. Calculated tours for passengers with high preferences against noise and smoke (left) versus high preferences for attractive areas (right) (Zipf, Roether, 1999)

III. COMMUNICATING WITH THE USER-MULTI-MODAL INTERACTION WITH LBS-THE EXAMPLE OF MAPS

After the user has specified his wishes regarding a tour and the tour has been calculated the next task is to present either the tour or a general overview map. This is the task of components that can generate dynamic maps and provide possibilities for interacting with these.

Here we see two main questions for GIScience regarding how

to improve communication between system and user about geographic phenomena. Their importance is stressed by the developments of UbiGIS fostering multiple representations and modalities on multiple devices with new interaction paradigms. So the more general research area here is the area of Human Computer Interaction (HCI), but not solely interaction and presentations issues need to be covered, but again the question of semantics.

Therefore the first-more general-question regards the representation of geospatial information and the semantic equivalence of these representations (Worboys, 2001). The

second one regards the “presentation” (in contrast to internal “representation”) of the spatial information to the user to support understanding and ease of use. This deals with personalized, contextualized representation (first of all visualization) as well as multi-modal interaction with geospatial information.

While research on supporting mobility, and in particular personalization and contextualization is relatively young in GIScience (Zipf, 1998), work on multi-modal interaction with spatial data is around for some time already (Oviatt, 1996). For example Schlaisisch and Egenhofer (2001) investigate an interaction method that mimics the natural communication between people and allows users to formulate queries. “Natural communication” includes here the combination of talking and sketching apart from traditional interaction methods. Other examples analyze the interaction with GIS on wall-scale displays (Hornsby, et al., 1996). This early work was mostly conceptual or empirical due to the limitations of technology at that date, but with the recent progress it becomes possible to develop systems that support interaction by speech or gestures etc. This allows testing the hypotheses posed, as well as developing new ones and evaluating these with actual systems (Schmidt-Belz, et al., 2002). This is certainly a step where GIScience needs to go, as a lot of questions on how to support situated interaction with multiple modalities for easing the use of geographical information are still open, lacking both a sound theoretical as well as an empirical underpinning.

In the following paragraphs we want to introduce the specifications for map generation both by OpenLS and the DMO specification and compare these— in particular with respect to the possibilities for realizing multi-modal applications based on these.

A. A comparison of map requests between openLS and the DMO specification

The DMO specification was also applied and extended within the SmartKom system (Bühler, et al., 2002), which supports different interaction modalities. The most important modalities for human-machine interaction considered in the SmartKom project are *speech recognition* and *synthesis*, *gesture recognition* (such as pointing on the screen or free gestures), *mimic recognition* using a face tracker, and *graphical display* of text, animations, and maps. Therefore the DMO Specification we developed had to be able to cope with the requirements from the scenarios including these interaction modes that should be supported within the system. The resulting DMO map request *GMAComputeMap* is shown, explained and compared to the OpenLS pendant *PortrayMapRequest* in Table 2.

GMAMapInteraction (Figure 9) is a GIS/Map agent action asking for the generation of an existing map along the given parameters. The map agent keeps a history of all maps it computed. These maps can be referenced with an identifier. The result of a map interaction is a complete new map with a new id. The advantage of the *GMAMapInteraction* request is

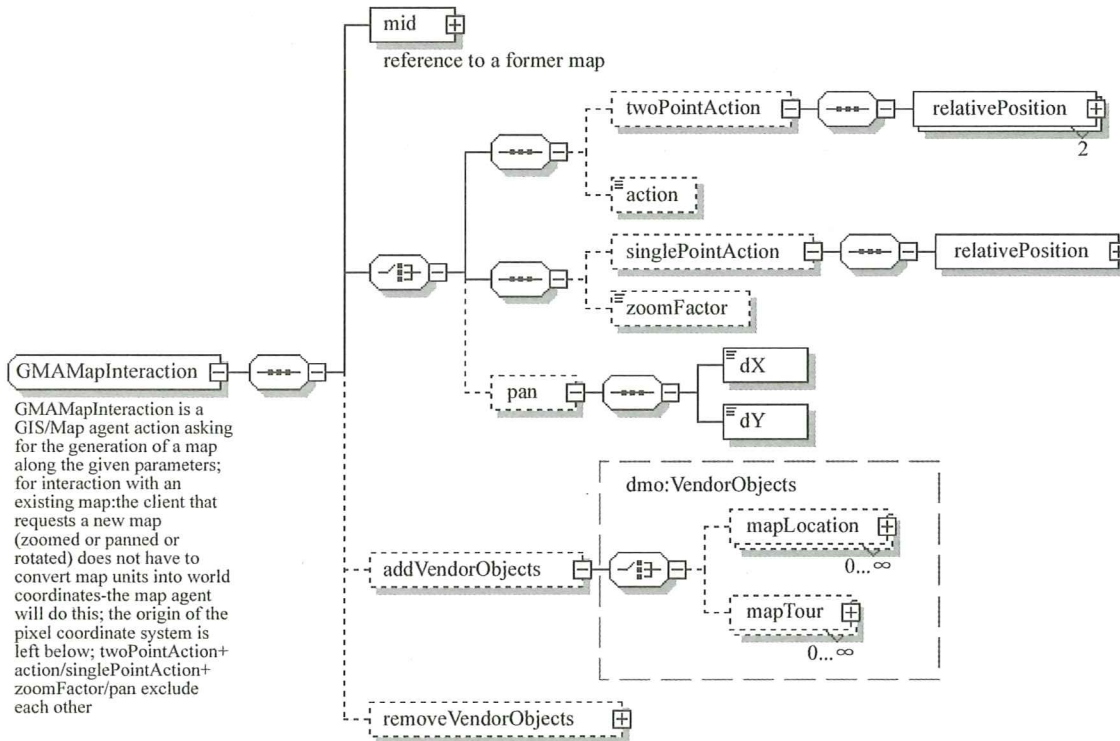


Figure 9. *GMA Map Interaction* is a specification for ‘client-friendly’ requests for map interactions. The client can specify interactions using pixel coordinates relative to the image origin. The translation to real world coordinates are done server side by the map agent

Table 2. Comparison of *portray map request* and *GMA compute map*

Functionalities	OpenLS:PortrayMapRequest	DMO:GMAComputeMap
<i>Map dimensions</i>	width, height	size (including width and height)
<i>Image encoding</i>	format (Attribute of Output, supported mime types depend on implementation)	imageFormat (enumeration-values: png, wbmp, jpg, gif, svg, svgCompressed, tif. not all types implemented)
<i>Background color</i>	BGColor	backgroundColor
<i>What type of content would be returned, whether URL or base64</i>	content (enumeration-values: URL, Data)	returnFormat (enumeration-values: base64, url)
<i>Definition of the area to be portrayed</i>	As bounding box or through definition of center and scale. In contrast to the deepmap interface, the screen resolution of the end device can be specified (DPI), which is necessary to derive the context from scale and map center. Rotated maps can be requested. If the map context is not specified the implementation must derive it from the objects to be overlaid.	extent. Can be specified as bounding box or as center-scale-box. Rotated maps can be requested. In contrast to the OpenLS Specification it can be defined how the map context has to be adapted relative to the overlaid objects. The element extentStrictness specifies if the extent has to be kept strictly or if it can be adapted to the vendor objects (and how). <i>exactly:</i> only adaption to the specified size is allowed. the given extent has to be shown completely in the map. <i>canGrow:</i> the extent must be enlarged if there are vendor objects outside the given box but it must not be shrunk. <i>canShrink:</i> the extent must be made smaller to spread the vendor objects over the map but it must not be enlarged if there are vendor objects outside the given box <i>canGrowAndShrink:</i> the map must be adapted to the vendor objects. In this case the given box is only relevant if there are no vendor objects.
<i>Objects to be portrayed on top of the base map</i>	Overlay. 0...n Points of Interest, route geometries, positions and maps can be overlaid to the requested map. With help of the <i>zorder</i> attribute, the order of the drawing of the objects can be specified explicitly.	vendorObjects. 0...n Locations and Tours can be overlaid. The objects have to be drawn in the same order as they occur in the request.
<i>Styling of the overlaid objects</i>	Named styles (well known by the server) or user defined styles can be specified for each object.	The overlaid locations and routes are contained in the request (and response) as MapLocation resp. MapTour, which are Locations resp. Tours plus styling information. Since e.g. tours are very complex structures in DMO it can be defined very detailed how to display the startLocation, the endLocation, the visitLocations and the route geometry (plus text labels).
<i>Definition of the layers of the base map plus styling information for these layers</i>	A list of Layers can be defined in Basemap . The attribute <i>filter</i> determines whether these layers are to be displayed or to be hidden. In the second case, all layers except the specified ones are to be displayed. Optionally for each layer a named or a user defined style can be specified. OpenLS Version 1.0 does not define a specific format for style description but has a placeholder for custom or future extensions.	The layers to be displayed on the map can be defined by a generalMapStyle. This is a definition known to the server resp. map agent which contains the list of layers plus styling information. Alternatively this list of layers plus styling information can be specified in the GMAComputeMap request as well.
<i>Manipulating maps</i>	If the user wants to manipulate a map (e.g. pan, zoom), a complete new PortrayMapRequest has to be sent.	For manipulating maps a client can send a GMAMapInteraction request to the mapping component.

that the map does not have to be specified again. So such a request would only specify the differences in appearance of the map to the former one. Table 3 shows two examples for *GMAMapInteraction* requests and illustrates how simple (and small sized) the messages are.

Figure 10 shows a simple way how a client application can generate a *GMAMapInteraction* request on basis of a circular

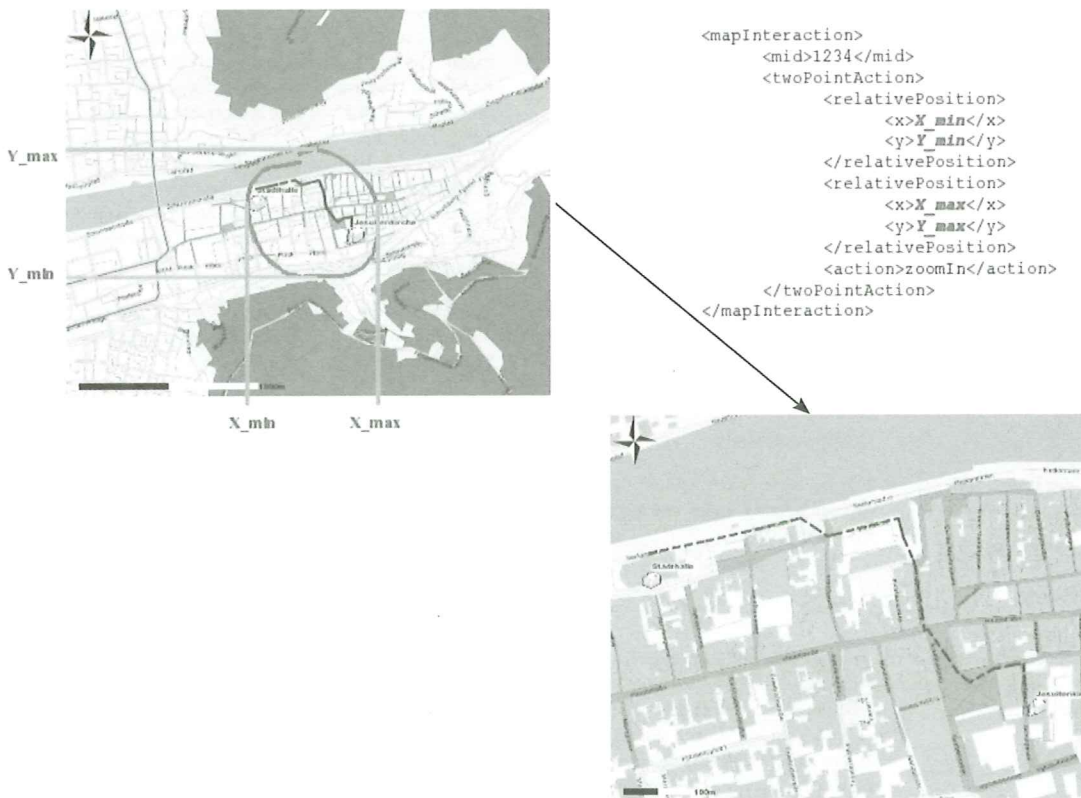
gesture by the user.

B. Representing maps in the response

The OpenLS *PortrayMapResponse* consists of one to many maps of the Map ADT. The Map ADT contains a rendered map that results from the Map Portrayal Operation of the Presentation Service. Parameters include format, width and

Table 3. Two example queries for map interaction of the DMO specification

<i>TwoPointAction with action "pan"</i>	<i>SinglePointAction</i>
<pre> <mapInteraction> <mid>123456</mid> <twoPointAction> <relativePosition> <x>50</x> <y>120</y> </relativePosition> <relativePosition> <x>150</x> <y>210</y> </relativePosition> <action>pan</action> </twoPointAction> </mapInteraction> </pre>	<pre> <mapInteraction> <mid>123457</mid> <singlePointAction> <relativePosition> <x>50</x> <y>120</y> </relativePosition> <zoomFactor>1.5</zoomFactor> </singlePointAction> </mapInteraction> </pre>

**Figure 10.** Realization of a circular gesture as a two pointAction for zooming

height, bounding box, center point and scale etc.

The DMO response to a *GMAComputeMap* and *GMAMapInteraction* is a *GRAvailableMap*. This is a map with rich information about all portrayed objects. All vendorObjects contain styling information. This is important metadata for more advanced clients like multi-modal dialog systems as SmartKom, where users may use this information to talk about the objects he beholds on the map. Since due to a pan or zoom interaction with a textual unmodified map some of the vendor objects may be outside the visible context of a map, the element *isInExtent* (for *MapLocations* and *MapTours*) gives the

information to the client, whether an object is completely or partially within the context of the map or not (enumeration-values: true, false, partially). The latter value makes sense for MapTours.

IV. SUMMARY AND OUTLOOK

Advantages of the OpenLS specification include on the technical side of the XML specification the more frequent use of attributes in contrast to elements. This leads to smaller and more simple structures that might be more suitable for wireless

transmission. However, as XML can be compressed very efficiently this is not a major issue. Relevant for the display of correct scale values for mobile maps is the possibility within the OpenLS map requests to specify display parameters like the dpi parameters of the device's display. This was planned but not realized in the DMO specification.

Regarding routes it seems sensible to be able to specify what information shall be returned as in the OpenLS route service. This eliminates the need to transmit information not needed in the current application. The DMO representation of routes on the other hand is more general and information-rich, but this might not be needed in all scenarios. The advantage of the DMO specification seems to be its richer expressiveness. This was gained through a more fine-grained modelling. Another big advantage is the additional power of the DMO *MapInteractionRequest* that allows more simple clients by hiding more complexity on the server. Further results of this are the much smaller requests for subsequent interaction with the same map. Only the first request for a map needs to include all the parameters specifying the contents of the map, while the following requests just reference that first one.

But in general both specifications provide similar functionality and we would have been happy if such a specification would have been available earlier. But on the other side we still believe that our modelling of some requests might act as input for further enhancements of the OpenLS specification – in particular when it comes to more sophisticated user-aware personalized tour planning (proposals) or multi-modal interaction with mobile GI services.

As a consequence of these (and other) new developments in mobile computing and Human Computer Interaction (HCI) and also through the ongoing development of interoperable spatial data infrastructures (SDI), one can expect GI services to be available ubiquitously to all users in the future. For this the term Ubiquitous Geographic Information Services = Ubiquitous GIS = UbiGIS has been suggested. This term can be defined as: Pervasive services based on UbiComp technology and devices, supporting context-dependent (i.e. adaptive) interaction, realized by information and functions of geographic information services based on interoperable SDI. It is being discussed by Reuter and Zipf (2004). If the visions come true, a platform will occur, that allows for utilizing different GI services from everywhere without any need to care for computer locations or networks. The desired simplicity of interaction shall ease the use of computerized services in general. In order to realize the possibilities outlined, still various research questions need to be worked on, in particular in the area of mobile maps (Meng, et al., 2004) and interaction. The comparison in this paper gives hints how to realize such ubiquitous GI services on the interface level based a.) on current standards (OpenLS) and b.) how to extend these in order to support adaptivity and multi-modal interaction as in the DeepMap or SmartKom systems.

ACKNOWLEDGEMENTS

The work presented in this paper was conducted within the SmartKom project partly funded by the German Ministry of Education and Research under grant 01IL9517 and by the Klaus Tschira Foundation and within the Deep Map project funded by the Klaus Tschira Foundation. We do thank all colleagues at EML and partners for their inspiration and help over the last years.

REFERENCES

- [1] Balas E. 1989. The Prize Collecting Traveling Salemans Problem. *Networks* 19. pp. 621–636.
- [2] Bühler D, Häußler J, Krüger S, Minker W. 2002. Flexible Multimodal Human-Machine Interaction in Mobile Environments. *Proceedings of the ECAI 2002 Workshop on Artificial Intelligence in Mobile System (AIMS)*, Lyon (Frankreich).
- [3] Häußler J, Jöst M, Zipf A. 2003. GIS Support for Multi-Modal Mobile Map Interaction. Poster at: COSIT 2003. *Conference on Spatial Information Theory*. pp. 24–28. Ittingen, Switzerland.
- [4] Hornsby J, Florence K, Egenhofer M. 1996. The GIS WallBoard: Interactions with Spatial Information on Large-Scale Displays. *Seventh International Symposium on Spatial Data Handling (SDH '96)*, Delft, The Netherlands. Kraak M J, Molenaar M (Eds.). pp. 8A.1–15.
- [5] Jöst M, Stille W. 2002. A User-Aware Tour Proposal Framework using a Hybrid Optimization Approach. *Proc. Of the 10th ACM International Symposium on Advances in Geographic Information Systems*. McLean, VA. ACM Press.
- [6] Kray C. 2003. *Situated Interaction on Spatial Topics*. PhD. thesis, DISKI series vol. 274.
- [7] Malaka R, Häußler J, Aras H, Merdes M, Pfisterer D, Jöst M, Porzel R. 2004. *Intelligent Interaction with a Mobile System*. Wahlster W(Ed.). SmartKom – Foundations of a Multimodal Dialogue System, Springer.
- [8] Meng L, Zipf A, Reichenbacher T(Eds.). 2004. *Map-based mobile services—Theories, Methods and Implementations*. Springer Geosciences. Springer Verlag. Heidelberg.
- [9] OGC. 2001. *The Geography Markup Language*. <http://www.opengeospatial.org/specs/>
- [10] OpenLS. 2003. *The OGC OpenLS Specification*. <http://www.openls.org/>
- [11] Oviatt S. 1996. *Multimodal Interfaces for Dynamic Interactive Maps*. CHI. *Conference on Human Factors in Computing Systems*. Vancouver, Canada.
- [12] Poslad S, Laamanen H, Malaka R, Nick A, Buckle P, Zipf A. 2001. CRUMPET: Creation of User-Friendly Mobile Services Personalised for Tourism. *Proceedings of: 3G 2001- Second Int. Conf. on 3G Mobile Communication Technologies*. pp.26–29. London. UK.
- [13] Rauschert I, Agrawal P, Sharma P, Fuhrmann S, Brewer I, Alan MacEachren. 2002. Designing a human-centered, multimodal GIS interface to support emergency management. *ACM GIS 2002. Proceedings of the tenth ACM international symposium on Advances in geographic information systems*. McLean, Virginia, USA. ACM Press. pp. 119–124.

- [14] Reuter A, Zipf A. 2004. GIScience – Where Next? Fotheringham S, Wilson J P(Eds.). Handbook of GIS. Blackwell Publishers.
- [15] Schlaisich I, Egenhofer M. 2001. Multimodal Spatial Querying: What People Sketch and Talk About. 1st International Conference on Universal Access in Human-Computer Interaction, New Orleans, LA, Stephanidis C(Editors). pp. 732–736.
- [16] Schmidt-Belz B, Zipf A, Poslad S, Laamen H. 2003. Location-based mobile tourist services - first user experiences. Int. Congress on Tourism and Communications Technologies. Helsinki. Finland. (Springer Computer Science. Heidelberg, Berlin).
- [17] Wahlster W(Ed.). 2004. SmartKom – Foundations of a Multimodal Dialogue System.
- [18] Weiser M. 1991. The Computer for the Twenty-First Century. Scientific American. pp. 94–104.
- [19] Zipf A, Röther S. 2000. Tourenvorschläge für Stadttouristen mit dem ArcView Network Analyst. Liebig J (Ed.). ArcView Arbeitsbuch. Hüthig. Heidelberg.
- [20] Zipf A, Aras H. 2002. Proactive Exploitation of the Spatial Context in LBS - through Interoperable Integration of GIS-Services with a Multi Agent System (MAS). Int. Conf. on Geographic Information Science of the Association of Geographic Information Laboratories in Europe (AGILE). Palma. Spain.
- [21] Zipf A. 1998. DEEP MAP - A prototype context sensitive tourism information system for the city of Heidelberg. Proc. of GIS-Planet 98, Lisbon, Portugal.
- [22] Zipf A. 2002. User-Adaptive Maps for Location-Based Services (LBS) for Tourism. Woeber K, Frew A, Hitz M(Eds.). Proc. of the 9th Int. Conf. for Information and Communication Technologies in Tourism, Innsbruck, Austria. Springer Verlag, Computer Science Series, Heidelberg.