Integration of Dynamic Atmospheric Modeling and Object-Oriented GIS*

Ludger Becker[1], Lars Bernard[2], Jürgen Döllner[1], Stefan Hammelbeck[1], Klaus H. Hinrichs[1], Thomas Krüger[2], Benno Schmidt[2] and Ulrich Streit[2]

Abstract

Efforts to integrate Geographic Information Systems (GIS) and environmental simulation models often fail due to the structure of current GIS which do not meet the requirements of complex numerical simulation techniques. Therefore research in the field of environmental modeling focuses on adequate flexible simulation methods and powerful interoperable GIS interfaces to implement these methods. This contribution shows approaches towards an interoperable object-oriented GIS interface for atmospheric modeling and analysis. The architecture and specifics of a first prototype system, called AtmoGIS, are presented.

1. Introduction

In general, GIS are not able to perform complex dynamic simulations due to lack of numerical capabilities, and environmental simulation systems do not support handling and analysis of geographic data. Therefore usually both geographic information systems (GIS) as well as dynamic environmental simulation systems are needed when modeling environmental processes (Fedra 1993). For exploring and analyzing spatio-temporal data it may even be necessary to couple these systems with a separate visualization system.

Today most interfaces between GIS and dynamic simulation systems are based on a strategy called *coupling* (Nyerges 1992). Each system uses its own data model, and data exchange requires a conversion between these data models. The same holds for many visualization systems which use data models optimized for 3D rendering. The coupling approach causes disadvantages like data redundancy, loss of data semantics, and inefficiency.

To avoid these disadvantages current research focuses on the development of frameworks enabling an *integration* of dynamic simulation models and GIS functionality (Bernard et al. 1998, Burrough and McDonnel 1998, Johnston et al. 1996) in such a way that all application components work on the same data and interact using the common methods provided by the frame-

* This project is founded by the German Science Foundation, DFG grant no. STR 172/8-1.

[1] Institut für Informatik, Westfälische Wilhelms-Universität Münster, Einsteinstr. 62, 48149 Münster,
email: {beckelu, dollner, hamstef, khh}@math.uni-muenster.de, http://wwwmath.uni-muenster.de

[2] Institut für Geoinformatik, Westfälische Wilhelms-Universität Münster, Robert Koch-Str. 26-28, 48149 Münster,
email: {bernard, benno, krugert, streit}@ifgi.uni-muenster.de, http://ifgi.uni-muenster.de

work. This approach results in two main requirements for the development of GIS and dynamic simulation systems:

1. Design and implementation of three-dimensional spatio-temporal data structures within GIS since current commercial GIS support only static two-dimensional data.
2. Design and development of interoperable software components which can be combined without creating new monolithic applications.

The development of open GIS environments which enable a time- and cost-efficient implementation of applications (Buehler und McKee 1996) is supported by the Open GIS Consortium (OGC 1998), an international not-for-profit membership organization which encourages GIS users and technology providers to develop technology standards in a consensus process.

Another initiative dealing with the design of interoperable systems called "Interoperable Offene Geowissenschaftliche Informationssysteme" (IOGIS 1997) is promoted by the German Science Foundation (DFG). IOGIS focuses on the integration of high-dimensional geoscientific simulation models with GIS and the utilization of object-oriented techniques for GIS development.

This contribution presents the object-oriented design and the prototypical implementation of an interoperable GIS environment which integrates a framework for the simulation of atmospheric processes (AtmoGIS).

2. AtmoGIS System Architecture

AtmoGIS provides a C++ class library which can be used to build interoperable components for managing and analyzing spatial data, for modeling atmospheric processes, and for visualizing spatio-temporal data sets. Figure 1 shows the architecture of the overall system.

AtmoGIS is based on an abstract, object-oriented “virtual GIS” layer (VirGIS), which allows AtmoGIS to communicate with the GIS kernel to handle grid-based spatio-temporal data. Currently, there exist two implementations of the GIS kernel: an object-oriented geo-database management system (GOODAC) and a simple, file-based system (FSGDM). This concept will be discussed in more detail in chapter 3.

Atmospheric simulation models (SKIMO, PDCA) are based on AtmoGIS classes which specify a simulation run’s characteristics and provide access to numerical solvers. The visualization framework MAM/VRS is used to perform 3D real-time rendering of the model results. Different user interface toolkits can be deployed to implement applications.

The interfaces between the system components can be realized in different ways, e.g., by blackbox abstraction or whitebox abstraction (Szyperski 1998). So far we are using the whitebox approach in order to take adavantage of object-oriented mechanisms such as subclassing, dynamic binding and templates.

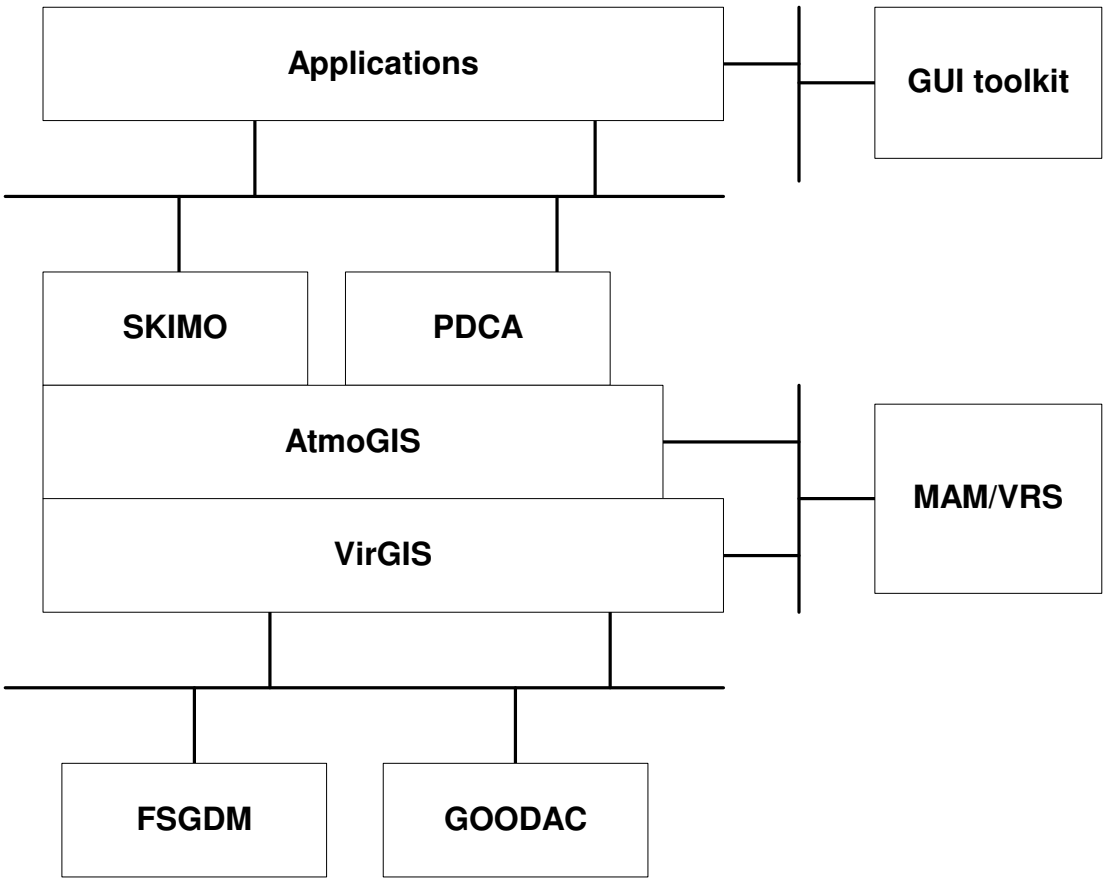

Figure 1: AtmoGIS architecture.

In the following the system components and their communication mechanisms will be described in more detail.

3. Interoperable Components of AtmoGIS

3.1 The Abstract Interface VirGIS

VirGIS provides as an interface a collection of abstract classes for managing spatio-temporal data, which are used for building environmental simulation models. Therefore it is possible to build interoperable applications based on VirGIS and GIS kernels which support the VirGIS interface.

Since grid-based data structures are of primary interest for building atmospheric simulation models VirGIS mainly provides classes for managing and processing high-dimensional grids. These grids can be traversed by so-called iterators. Grid data can be accessed directly using grid indices or through a set of spatio-temporal interpolation methods.

3.2 VirGIS Implementations Using an Object-oriented Geo-Database

One of the two current VirGIS implementations is based on FSGDM (File System Geo Data Management). Another VirGIS implementation is based upon the Object-Oriented Geo Data Model (OOGDM) and its prototypical implementation GOODAC (Geo Object-Oriented Database Core). In contrast to FSGDM, GOODAC supports database management features such as transactions, integrity constraints, or a query language.

OOGDM is an object-oriented data model which has been designed to form the basis of GIS applications. OOGDM defines a hierarchy of classes covering most types of spatial data found in GIS applications and supports raster- and vector-based data in 2D and 3D space. Furthermore, OOGDM can handle time-varying data by incorporating concepts from temporal databases. Further details can be found in (Voigtmann 1998). Basically, OOGDM distinguishes between two (orthogonal) dimensions of time: *Valid-time* describes the validity of real world entities, whereas *transaction-time* describes the validity of entities in the database. Commonly used spatial operations are supported by the OOGDM-classes.

Associated with OOGDM are an Object Definition Language OOGDM-ODL and the SQL-like query language OOGQL (Object-Oriented Geo Query Language). OOGDM-ODL supports the creation of new classes for a database in a C++-like style. The object definition language satisfies the individual modeling needs of GIS application developers: Developers may extend the data model by deriving application dependent classes from the predefined class hierarchy of OOGDM or by defining classes independently from OOGDM.

Queries to an OOGDM-based database can be formulated in the SQL-like query and manipulation language OOGQL. The design of OOGDM-ODL and OOGQL resembles and extends the languages in the *Object Database Standard* (Catell 1996) by concepts for spatial and temporal data (Snodgrass 1995).

3.2.1 GOODAC: A Prototypical Implementation of OOGDM

GOODAC (Becker et al. 1996, Voigtmann et al. 1997) is an extensible GIS database core, which realizes the OOGDM class hierarchy. GOODAC is basically realized by a two-layer architecture, which is shown in figure 2. The top layer is the descriptive layer, which is the view an application developer and a user of the system have. The data model of this layer is OOGDM. The bottom layer of GOODAC is the representation layer. The data model of this layer is the so-called representation data model consisting of representations for the geometry of OOGDM-objects, implementations for geometric and topological operations, index structures, and stream based query processing methods. To meet the requirements of various GIS-application areas, GOODAC has been designed as an extensible system.

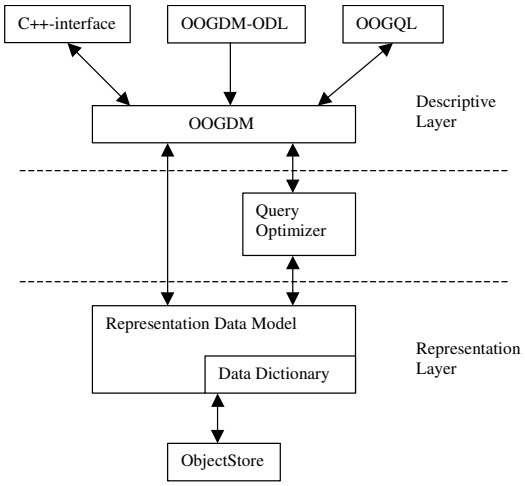

Figure 2: GOODAC system architecture

The representation layer has been implemented on top of the object-oriented database system ObjectStore (Lamb et al. 1991). ObjectStore provides support for object storage, multi-user access, and transaction management.

3.2.2 Interoperability Between Implementation Objects and Application Layer

Since AtmoGIS objects are based on VirGIS objects, which are themselves realized by OOGDM- or FSGDM-objects, AtmoGIS operates directly on the persistent data structures. As a result of the system architecture, a real integration is achieved between AtmoGIS, VirGIS, and the VirGIS implementations. This complete integration of an application and a GIS avoids the loss of semantics occurring in the traditional approach since model specific conversions and exchange formats are no longer necessary. VirGIS realizes interoperability with respect to the underlying GIS. VirGIS based applications (like AtmoGIS) do not have to be aware which VirGIS implementation they use. The application development only depends on the abstract interface.

3.3 AtmoGIS

For the development of AtmoGIS different simulation strategies used in the context of meso- and microscale atmospheric modeling have been analyzed with respect to their requirements. AtmoGIS offers several base classes to realize atmospheric simulation models and to implement applications which integrate models and analysis tools. In its current development stage AtmoGIS contains

- classes to manage spatio-temporal meteorological fields,
- classes to implement pre- and postprocessing tools, and

- simulation base classes to support time management, model control and automatic simulation documentation.

The AtmoGIS classes specify the basic behavior of the integrated simulation models which are considered as simulation engines managing several simulators. These simulators solve physical equations, calculate boundary values, provide empirical values for the simulation run and manage simulation results using the abstract VirGIS interface. Complex simulators aggregate several simulators and therefore support the design of model hierarchies. Simulators can be exchanged between different simulation engines. This interoperability of simulators enables model developers to reuse simulators. Basic simulators provided by AtmoGIS support typical boundary value calculations, the handling of terrain following coordinates, model documentation, and offer some grid-based numerical solvers (see Krüger and Bernard 1999 for a detailed description).

The object-oriented simulator concept allows the application developer to formulate constraints which control *semantic correctness* of either a simulation run or an analysis process (see Bernard and Pundt, 1998). For instance the physical requirements of a simulator can be expressed (e.g.: *This simulator requires the assumption of a hydrostatic atmosphere*) or the minimal reasonable spatial resolution for an overlay with the simulation results can be constrained (e.g.: *Cellsize for this analysis should be > 100 m*). Using AtmoGIS two different simulation models have been integrated:

- S_KIMO-3 is a prognostic boundary layer model based on the equations of motion assuming hydrostatic and incompressible flows.
- PDCA (Pollutant Dispersion Cellular Automata) is a microscale dispersion model based on cellular automata.

Different prototype applications using these models will be developed to prove the claimed interoperability.

3.4 The Interoperable Visualization System MAM/VRS

Many powerful techniques for visualizing geo-data and their dynamics have been developed with the goal to provide insight into the data and to understand the dynamics of the data. In general, application developers use high-level toolkits, the so-called visualization toolkits (e.g., Schroeder et al. 1996), which provide components for presenting, exploring, and analyzing data.

An interoperable system architecture requires that the visualization system itself provides the necessary interoperability without sacrificing efficiency which is required for real-time rendering. Furthermore the visualization system must be extendible in order to meet the specific visu-

alization needs of concrete GIS applications. In our approach, MAM/VRS (Döllner and Hinrichs 1997) has been used as interoperable visualization system for AtmoGIS.

MAM/VRS provides components for modeling virtual 3D scenes based on the well-known scene-graph paradigm, and components for modeling time-dependent and event-dependent behaviors based on so-called behavior graphs which permit a compact specification of computer animations and human-computer interactions. Different 3D rendering systems, for example OpenGL for real-time rendering and RenderMan for photorealistic rendering, can be used by MAM/VRS. The criteria for the interoperable deployment of MAM/VRS are discussed in the following; for a detailed analysis see (Döllner and Hinrichs 2000).

3.4.1 Use of Application Data Structures for Rendering Primitives

The visualization components, in particular the rendering primitives of MAM/VRS, do not require a specific internal data structure: they are able to operate on application-specific data structures. This mechanism has been implemented by *iterators*, which import and export the necessary data from and to the application on demand.

An iterator is an object that provides sequential access to application data. Any application data structure to be used within MAM/VRS must be wrapped by an iterator class. The MAM/VRS visualization components operate exclusively at the iterator level – no specific knowledge of the underlying data structure is necessary.

For example, the height-field visualization component defined by a 2D array of 3D points is specified by four iterators, a vertex iterator, a vertex-color iterator, a vertex-normal iterator, and a vertex-texture coordinates iterator. An application developer might decide to implement the vertex iterator by database queries, the vertex-normal iterator analytically, and the color-iterator procedurally.

The iterator approach has been motivated by the fact that in most cases the GIS data representation is different from the graphics representation. In contrast to data conversion used in the coupling approach, iterators permit a functional and dynamic binding of both representations. Conversion is performed on demand, and no redundant storage in an intermediate format is required. In addition, the semantics is maintained in the visualization representation because iterators have the necessary access to the data sources.

3.4.2 Integration of MAM/VRS with AtmoGIS

AtmoGIS uses the visualization system MAM/VRS to present, explore and analyze atmospheric data and their dynamics. AtmoGIS operates basically on multi-dimensional grids which are

linked to MAM/VRS components by iterator classes. Specific components for visualization flows have been added to MAM/VRS.

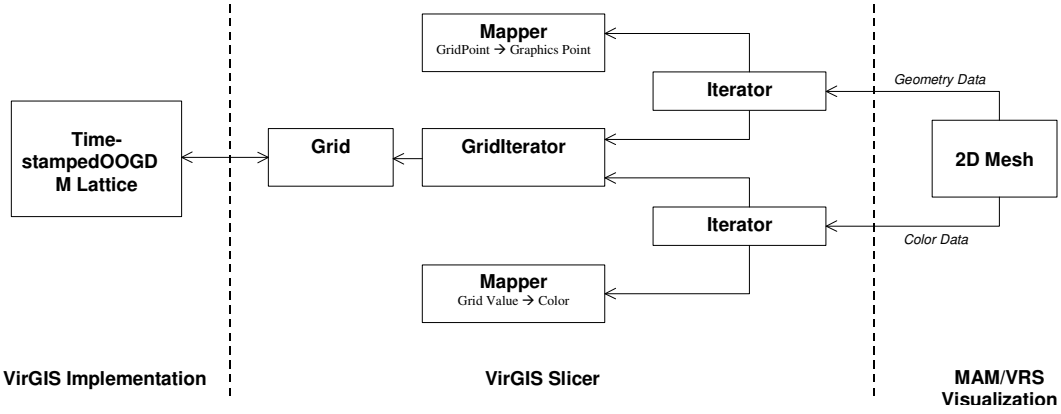

Figure 3: Implementation of a data slicer with AtmoGIS and MAM/VRS.

The class diagram in figure 3 illustrates the implementation of a data slicer in AtmoGIS. The grid represents temperature values contained in a 3D grid. The slicer is represented by a planar graphics mesh which can be swept through the data volume. The mesh cells visualize interpolated temperature values of the grid according to a coloring scheme. A grid iterator represents the iterator for the grid data structure. The grid used in this example is a time-variant 3D grid implemented by the OOGDM system. The mappers are used to map grid data types to graphics data types. Database and visualization are linked permanently by the iterators which are used each time the mesh is rendered. In order to improve rendering efficiency the visualization system can decide to cache visualization objects unless they are modified. Based on the interface classes provided by VirGIS, visualization techniques can be implemented which can be used independently from the underlying VirGIS implementation.

Figure 4 shows a screenshot of an application built on top of the AtmoGIS components. The visualized atmospheric data result from a simulation run of the model S_KIMO-3. The three-dimensional data of a single time-step are displayed on a horizontal slicer in case of the wind field respectively on a vertical slicer in case of the temperature field. The application successfully integrates the S_KIMO-3 model with AtmoGIS and MAM/VRS, using VirGIS.

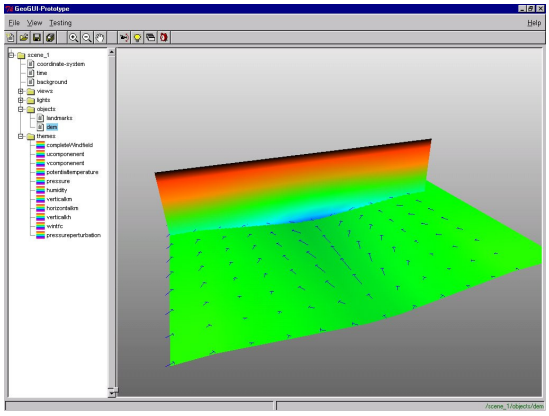

Figure 5: End-user application showing S_KIMO-3 result data.

4. Further Issues

The results of this work lead to the integrated use of dynamic environmental simulation systems and GIS. Related works include the integration of AtmoGIS in the desktop GIS ArcView and the realization of these ideas using distributed computing platforms (DCP). It will be a challenge to integrate distributed services for accessing geo-data (e.g., OpenGIS Catalog Services; OGC 1999) and for processing geo-data with distributed simulation services (e.g., HLA; DoD 1998).

References

Becker, L., Voigtmann, A., Hinrichs, K.H. (1996): *Developing Applications with the Object-Oriented GIS-Kernel GOODAC*, Proc. 7th Int. Symp. On Spatial Data Handling (SDH'96), Delft, The Netherlands, pp. 5A.1 – 5A.18.

Bernard, L., Pundt, H. (1998): *Semantikverlust in integrierten Systemen – Ein Fallbeispiel aus der Umweltplanung*, in: Umweltinformatik 98, 12. Internationales Symposium "Informatik für den Umweltschutz", Bd. 2, Marburg, pp. 528-540.

Bernard, L., Schmidt, B., Streit, U., Uhlenküken, C. (1998): *Managing, Modeling, and Visualizing High-Dimensional Spatio-Temporal Data in an Integrated System.* GeoInformatica, Vol. 2, No. 1, Norwell, MA: Kluwer Academic Publishers, pp. 59-77.

Buehler, K., McKee, L. (1996): *The OpenGIS Guide - Introduction to Interoperable Geoprocessing.* Technical report, Open Geodata Interoperability Specification (OGIS), Open GIS Consortium, Inc.

Burrough, P.A., Mc.Donnel, R.A. (1998): *Principles of Geographical Information System*, Oxford.

Catell, R.G.G. (1996): *The Object Database Standard ODMG-93, Release 1.2*, Morgan-Kaufmann Publishers, San Francisco, CA.

DoD (1998): *High Level Architecture Interface Specification (Version 1.3)*, U.S. Department of Defense, http://hla.dmso.mil.

Döllner, J., Hinrichs, K.H. (1997): *Object-oriented 3D Modeling, Animation, and Interaction. The Journal of Visualization and Computer Animation*, 8(1), pp. 33-64.

Döllner, J., Hinrichs, K.H. (2000): *An Object-Oriented Approach for Integrating 3D Visualization Systems and GIS.* Computer & Geosciences, special issue "Geoscientific Visualization", Elsevier Science, 26(1).

Fedra, K. (1993): *GIS and Environmental Modeling*, in: Goodchild, M. F., Parks, B. O., and Steyaert, L. T. (eds.): Environmental Modeling with GIS, Oxford University Press, Oxford.

IOGIS (1997): *Interoperable Offene Geowissenschaftliche Informationssysteme (IOGIS)*, http://ifgi.uni-muenster.de/3_projekte/4dgis/texte/iogis/IOGIS.html.

Johnston, C.A., Cohen, Y., Pastor, J. (1996): *Modeling of Spatially Static and Dynamic Ecological Process*, in: Goodchild et al. (Hrsg.): GIS and Environmental Modelling, GIS World Books, Fort Collins, pp. 149-154.

Krüger, T., Bernard, L. (1999): *Integration meteorologischer Ausbreitungsmodelle in GIS - Interoperable und objektorientierte Ansätze für verschiedene Modellstrategien*, in: Grützner R. (Hrsg.): Werkzeuge für die Modellierung und Simulation im Umweltbereich. Metropolis, Marburg, pp. 179-193.

Lamb, C., Landis, G., Orenstein, J., Weinreb, D. (1991): *The ObjectStore Databse System*, Communications of the ACM, 34 (10), pp. 50 – 63.

Nyerges, T.L. (1992): *Coupling GIS and Spatial Analytic Methods*, Proc. 5th Int. Symposium on Spatial Data Handling, pp. 534-543.

Schroeder, W.J., Martin, K.M., Lorensen, W.E. (1996): *The Design and Implementation of an Object-Oriented Toolkit for 3D Graphics and Visualization*, IEEE Proc. Visualization'96, pp. 93 – 100.

Snodgrass, R.T. (1995): *The TSQL-2 Temporal Query Language*, Kluwer Academic Publishers, Dordrecht, The Netherlands.

Szyperski, C. (1998): *Component Software – Beyond Object-Oriented Programming*, Addison-Wesley, New York.

OGC (1998): *Open GIS Consortium*, http://www.opengis.org.

OGC (1999): *OpenGIS – Catalog Interface Implementation Specification (Version 1.0)*, OpenGIS Project Document 99-051s, Open GIS Consortium, http://www.opengis.org.

Voigtmann, A., Becker, L., Hinrichs, K.H. (1997): *Physical Design Aspects of Object-Oriented Geo Database Kernel*, Proc. 8th Int. Workshop on Database and Expert Systems Applications (Workshop DEXA'97), Toulouse, France, pp. 529 – 534.

Voigtmann, A. (1998): *An Object-Oriented Database Kernel for Spatio-Temporal Geo-Applications*, Inaugural-Dissertation, Institut für Informatik der Westfälischen Wilhelms-Universität Münster.